W3School Foundation 教程

wizardforcel

Published with GitBook



目錄

W3School Foundation 教程

作者：W3School

来源：Foundation 教程

Foundation 入门

Foundation 5 简介



Foundation 用于开发响应式的 HTML, CSS and JavaScript 框架。

Foundation 是一个易用、强大而且灵活的框架,用于构建基于任何设备上的 Web 应用。

Foundation 是一个以移动优先的流行框架。

在线实例

菜鸟教程包含了上百个 Foundation 实例。

你可以直接使用我们的在线编辑器，并点击"提交运行"按钮查看结果：

Foundation 实例

```
<div class="row">
  <div class="medium-12 columns">
    <div class="panel">
      <h1>Foundation Page</h1>
      <p>Resize this responsive page to see the effect!</p>
      <button type="button" class="button small">I Like It!</button
    </div>
  </div>
</div>

<div class="row">
  <div class="medium-4 columns">
    <h3>Column 1</h3>
    <p>Lorem ipsum..</p>
  </div>
  <div class="medium-4 columns">
    <h3>Column 2</h3>
    <p>Lorem ipsum..</p>
  </div>
  <div class="medium-4 columns">
    <h3>Column 3</h3>
    <p>Lorem ipsum..</p>
  </div>
</div>
```

点击"尝试一下"按钮查看在线实例。

Foundation 特性

以下效果为在 iframe 标签中的演示，可以点击"尝试一下"查看在线实例：

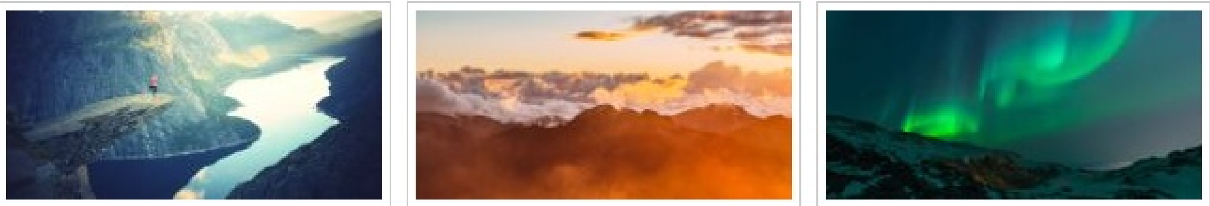
按钮



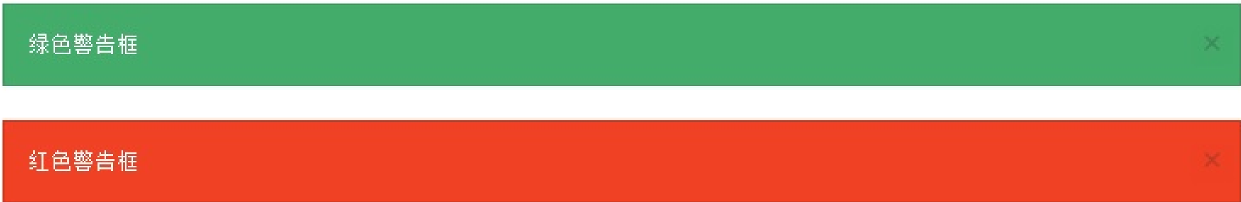
表格:

Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

图片弹窗:



警告:



网格:

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1

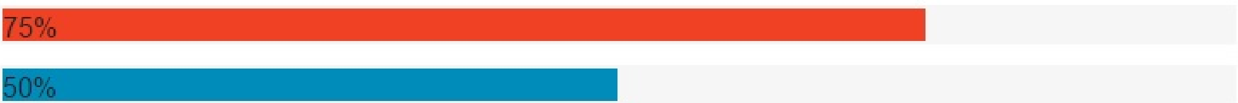
span 4 | span 4 | span 4

span 4 | span 8

span 6 | span 6

span 12

进度条:



面板:

Gray Panel

Panel Text

Blue Panel

Panel Text

下拉菜单:

Dropdown Button

Link 1

Link 2

Link 3

手风琴效果:

Accordion 1

Demo 1 - Lorem ipsum dolor sit amet....

Accordion 2

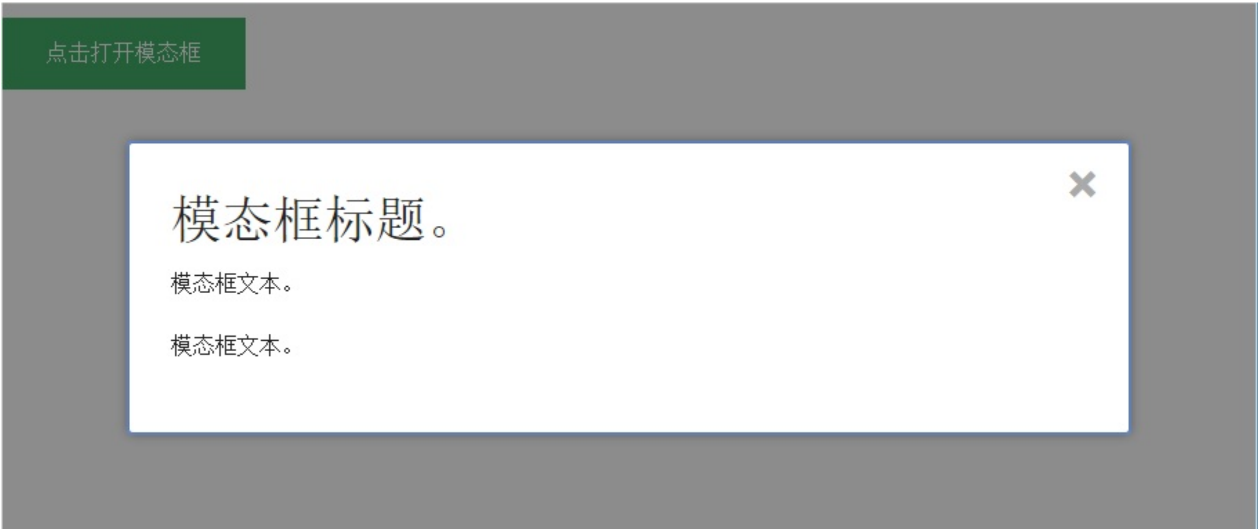
Accordion 3

>

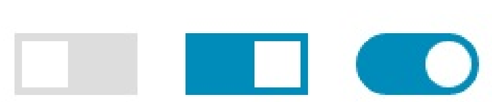
顶部导航:

Home Sign Up Search

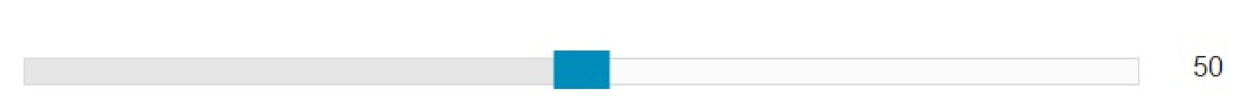
模态框:



开关:



滑块:



Foundation 起步

什么是 Foundation？

- Foundation 是一个免费的前端框架，用于快速开发。
- Foundation 包含了 HTML 和 CSS 的设计模板，提供多种 Web 上的 UI 组件，如表单、按钮、Tabs 等等。同时也提供了多种 JavaScript 插件。
- Foundation 移动优先，可创建响应式网页。
- Foundation 适用于初学者和专业人士。
- Foundation 已使用在 Facebook, eBay, Samsung, Amazon, Disney等。

> 
>
> 什么是响应式网页设计？
>
> 响应式Web设计(Responsive Web design)的理念是：页面的设计与开发应当根据用户行为以及设备环境(系统平台、屏幕尺寸、屏幕定向等)进行相应的响应和调整。

从哪里下载 Foundation?

你可以通过以下两种方式来获取 Foundation：

1、从官网下载最新版本：http://foundation.zurb.com/。

2、使用菜鸟教程官网提供的CDN（推荐）：

```
<!-- css 文件 -->
<link rel="stylesheet" href="http://static.runoob.com/assets/founda

<!-- jQuery 库 -->
<script src="http://static.runoob.com/assets/jquery/2.0.3/jquery.mi

<!-- JavaScript 文件 -->
<script src="http://static.runoob.com/assets/foundation-5.5.3/js/fo

<!-- modernizr 文件 -->
<script src="http://static.runoob.com/assets/foundation-5.5.3/js/ve
```

本站静态 CDN 基于阿里云服务。



Foundation 使用 CDN 的优势:

Foundation 使用 CDN 提高了企业站点(尤其含有大量图片和静态页面站点)的访问速度，并大大提高以上性质站点的稳定性

为什么使用 **modernizr?**

一些 Foundation 的组件使用了比较前前沿的 HTML5 和 CSS3 特性，但不是所有浏览器都支持。 Modernizr 是一个用于检测用户浏览器HTML5和CSS3特性的JavaScript库 - 让组件能在所有浏览器上正常运行。

使用 Foundation 创建页面

1. 添加 HTML5 doctype

Foundation 使用 HTML 元素和 CSS 属性，所以需要添加 HTML5 doctype 文档类型声明。

同时我们可以设置文档的语言属性 lang 及字符编码：

```
<!DOCTYPE html>
<html lang="zh-cn">
  <head>
    <meta charset="utf-8">
  </head>
</html>
```

2. Foundation 5 移动优先

Foundation 5 为移动设备的响应式设计。框架的核心是移动优先。

为了确保页面可自由缩放可以在 `<head>` 元素中添加以下 `<meta>` 标签:

```
<meta name="viewport" content="width=device-width, initial-scale=1'
```

- width：控制 viewport 的大小，可以指定的一个值，如果 600，或者特殊的值，如 device-width 为设备的宽度（单位为缩放为 100% 时的 CSS 的像素）。
- initial-scale：初始缩放比例，也即是当页面第一次 load 的时候缩放比例。

3. 初始化组件

一些 Foundation 组件是基于 jQuery 开放的，如：模态框、下拉菜单等。你可以使用以下脚本来初始化组件：

```
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

基本 Foundation 页面

如何创建一个基本的 foundation 页面:

```
<!DOCTYPE html>
<html>
<head>
  <title>Foundation Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=
  <!-- css 文件 -->
  <link rel="stylesheet" href="http://static.runoob.com/assets/foun

  <!-- jQuery 库 -->
  <script src="http://static.runoob.com/assets/jquery/2.0.3/jquery.

  <!-- JavaScript 文件 -->
  <script src="http://static.runoob.com/assets/foundation-5.5.3/js/

  <!-- modernizr 文件 -->
  <script src="http://static.runoob.com/assets/foundation-5.5.3/js/
</head>
<body>

<div class="row">
  <div class="medium-12 columns">
    <div class="panel">
      <h1>Foundation 页面</h1>
      <p>重置窗口大小，查看效果！</p>
      <button type="button" class="button small">I Like It!</button
    </div>
  </div>
</div>

<div class="row">
  <div class="medium-4 columns">
    <h3>Column 1</h3>
    <p>Lorem ipsum lorem ipsum lorem ipsum lorem ipsum lorem ipsum
  </div>
  <div class="medium-4 columns">
    <h3>Column 2</h3>
    <p>Lorem ipsum lorem ipsum lorem ipsum lorem ipsum lorem ipsum
  </div>
  <div class="medium-4 columns">
    <h3>Column 3</h3>
    <p>Lorem ipsum lorem ipsum lorem ipsum lorem ipsum lorem ipsum
  </div>
</div>

</body>
</html>
```

Foundation 文本

Foundation 默认设置

Foundation 使用浏览器默认字体大小 (font-size:100%)。对于大多数桌面浏览器来说，字体默认为 16px。对于大多数移动端浏览器来说，字体默认为 12px。默认的字体为 "Helvetica Neue" ， line-height 默认为 1.5 。

以上默认的设置均是针对 <body> 元素。

Foundation 文字排列设计

本章节我们将讨论 Foundation 的文字排列设计。

以下实例的真实样式请通过点击"尝试一下"按钮查看。

<h1> - <h6>

Foundation 渲染的 HTML 标题 (<h1> 到 <h6>) 如下所示:

实例

```
<h1>h1 标题</h1>
<h2>h2 标题</h2>
<h3>h3 标题</h3>
<h4>h4 标题</h4>
<h5>h5 标题</h5>
<h6>h6 标题</h6>
```

提示： 如果需要创建一个浅色的标题，可以在元素上添加 .subheader 类:

实例

```
<h1 class="subheader">h1.subheader</h1>
<h2 class="subheader">h2.subheader</h2>
<h3 class="subheader">h3.subheader</h3>
<h4 class="subheader">h4.subheader</h4>
<h5 class="subheader">h5.subheader</h5>
<h6 class="subheader">h6.subheader</h6>
```

<small>

在 Foundation 中， HTML <small> 元素用于创建一个浅色的副标题：

实例

```
<h1>h1 heading <small>secondary text</small></h1>
<h2>h2 heading <small>secondary text</small></h2>
<h3>h3 heading <small>secondary text</small></h3>
<h4>h4 heading <small>secondary text</small></h4>
<h5>h5 heading <small>secondary text</small></h5>
<h6>h6 heading <small>secondary text</small></h6>
```

<a>

Foundation <a> 元素的样式如下所示：

实例

```
<p>这是一个 <a class="a" href="#">链接</a>。</p>
```

<abbr>

Foundation <abbr> 元素的样式如下所示：

实例

```
<p>The <abbr title="World Health Organization">WHO</abbr> was foun
```

<blockquote>

Foundation <blockquote> 元素的样式如下所示：

实例

```
<blockquote>
  <p>学的不仅是技术，更是梦想！</p>
  <cite>菜鸟教程</cite>
</blockquote>
```

<dl>

Foundation `<dl>` 元素的样式如下所示：

实例

```
<dl>
        <dt>Coffee</dt>
        <dd>- black hot drink</dd>
        <dt>Milk</dt>
        <dd>- white cold drink</dd>
      </dl>
```

<code>

Foundation `<code>` 元素的样式如下所示：

实例

```
<p>以下 HTML 元素: <code>span</code>, <code>section</code>, 和 <code
```

<kbd>

Foundation `<kbd>` 元素的样式如下所示：

实例

```
<p>按下 <kbd>ctrl + p</kbd> 键打开打印窗口。</p>
```

<hr>

Foundation `<hr>` 元素的样式如下所示：

实例

```
<hr>
```

Foundation 表格

Foundation 的 `<table>` 元素样式为灰色斑马条纹且包含四个边框：

Firstname	Lastname	Email	City	Age	Country
John	Doe	john@example.com	New York	35	USA
Mary	Moe	mary@example.com	Chicago	51	USA
July	Dooley	july@example.com	San Francisco	38	USA

实例

```
<table>
  <thead>
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Email</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>John</td>
      <td>Doe</td>
      <td>john@example.com</td>
    </tr>
    <tr>
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.com</td>
    </tr>
    <tr>
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
    </tr>
  </tbody>
</table>
```

响应式表格

使用 CSS 让表格支持响应式设计：在表格外添加 `<div>` 元素，样式为 `overflow-x:hidden`：

实例

```
<div style="overflow-x:hidden">
  <table>
    ...
  </table>
</div>
```

Foundation 按钮

按钮样式

Foundation 提供了 6 种按钮样式。 .button 类创建了一个蓝色的按钮并附有内边距。不同颜色按钮类为： .secondary , .success , .info , .warning 或 .alert :

实例

```
<button type="button" class="button">Default</button>
<button type="button" class="button secondary">Secondary</button>
<button type="button" class="button success">Success</button>
<button type="button" class="button info">Info</button>
<button type="button" class="button warning">Warning</button>
<button type="button" class="button alert">Alert</button>
```

按钮类可以使用在 <a> , <button> , 或 <input> 元素:

实例

```
<a href="#" class="button info" role="button">Link Button</a>
<button type="button" class="button info">Button</button>
<input type="button" class="button info" value="Input Button">
<input type="submit" class="button info" value="Submit Button">
```

> 
>
> 为什么将 **a** 标签的 **href** 设置为 **#？**
>
> 当我们不希望链接点击跳转并得到 "404" 页面时，我们可以将 a 标签的 href 设置为 #。

按钮大小

我们可以使用 .large , .small 或 .tiny 类来设置按钮大小：

实例

```
<button type="button" class="button large">Large</button>
<button type="button" class="button">Default</button>
<button type="button" class="button small">Small</button>
<button type="button" class="button tiny">Tiny</button>
```

圆角按钮

可以使用 `.radius` 和 `.round` 类来设置圆角按钮：

实例

```
<button type="button" class="button">Default Button</button>
<button type="button" class="button radius">Radius Button</button>
<button type="button" class="button round">Round Button</button>
```

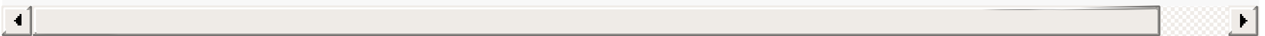


延展按钮

可以使用 `.expand` 类来设置按钮的宽带为 100%:

实例

```
<button type="button" class="button">Default Button</button>
<button type="button" class="button expand">Expanded Button</buttor
```

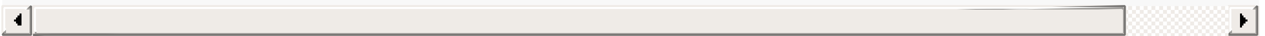


禁用按钮

可以使用 `.disabled` 类来设置按钮不可点击:

实例

```
<button type="button" class="button">Default Button</button>
<button type="button" class="button disabled">Disabled Button</butt
```



Foundation 按钮组

按钮组

Foundation 可以在同一行内创建一系列的按钮。

可以使用 `<ul>` 元素并添加 `.button-group` 类来创建按钮组：

实例

```
<ul class="button-group">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>
```

垂直按钮组

垂直按钮组使用 `.stack` 类来创建。注意，按钮会跨越父元素的整个宽度:

实例

```
<ul class="button-group stack">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>
```

`.stack-for-small` 类用于小尺寸的屏幕，按钮有水平排列变为垂直排列：

实例

```
<ul class="button-group stack-for-small">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>
```

圆角按钮组

按钮组中使用 `.radius` 和 `.round` 类为按钮添加圆角效果：

实例

```
<ul class="button-group radius">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>

<ul class="button-group round">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>
```

均匀延展按钮组

`.even-num` 类用于在按钮组中均匀的分配按钮的宽度并跨越父元素 100% 宽度。*num* 为按钮组中按钮的数量，从 1 到 8：

实例

```
<ul class="button-group even-3">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
</ul>

<ul class="button-group even-5">
  <li><button type="button" class="button">Apple</button></li>
  <li><button type="button" class="button">Samsung</button></li>
  <li><button type="button" class="button">Sony</button></li>
  <li><button type="button" class="button">HTC</button></li>
  <li><button type="button" class="button">Huawei</button></li>
</ul>

<ul class="button-group even-8">
  <li><button type="button" class="button">A</button></li>
  <li><button type="button" class="button">B</button></li>
  <li><button type="button" class="button">C</button></li>
  <li><button type="button" class="button">D</button></li>
  <li><button type="button" class="button">E</button></li>
  <li><button type="button" class="button">F</button></li>
  <li><button type="button" class="button">G</button></li>
  <li><button type="button" class="button">H</button></li>
</ul>
```

下拉菜单按钮

下拉菜单按钮可以让用户选取设定好的值：

实例

```
<!-- Trigger the dropdown -->
<a href="#" data-dropdown="id01" class="button dropdown">Dropdown B

<!-- The actual dropdown -->
<ul id="id01" data-dropdown-content class="f-dropdown">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

实例解析

`.dropdown` 类创建一个下拉菜单按钮。

使用带有 `data-dropdown="_id_"` 属性的按钮或链接打开下拉菜单。

id 值需要与下拉菜单的内容 (id01) 匹配。

在 `<ul>` 中添加 `.f-dropdown` 类和 `data-dropdown-content` 属性来创建下拉菜单的内容。

最后初始化 Foundation JS。

分割按钮

我们也可以创建一个分割按钮的下拉菜单。只需要在按钮中添加 `.split` 类并使用 span 元素生成一个方向箭的按钮：

实例

```
<button class="button split">Split Button
  <span data-dropdown="id01"></span>
</button>

<ul id="id01" data-dropdown-content class="f-dropdown">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```



提示:后面的教程中我们将学到更多关于下拉菜单是知识。

Foundation 图标

Foundation 提供了 283 个图标，你可以使用css来定义它的样式尺寸。

图标可用于文本，按钮，工具条，导航栏，表单等。

以下是 Foundation 图标的实例：

刷新按钮:

检测图标:

主页图标:

图标语法

创建图标语法格式如下:

```
<i class="fi-name"></i>
```

name 部分替换为图标的名字。

要使用图标我们需要在 <head> 部分添加图标的样式文件：

```
<link rel="stylesheet" href="http://static.runoob.com/assets/founda
```

Icon 实例

以下演示了使用图标的实例:

实例

```
<p>Cloud icon: <i class="fi-cloud"></i></p>
<p>Cloud icon as a link:
  <a href="#"><i class="fi-cloud"></i></a>
</p>
<p>Styled Cloud icon: <i class="fi-cloud" style="font-size:35px;col
<p>Home icon: <i class="fi-home"></i></p>
<p>Home icon on a button:
  <button type="button" class="button">
    <i class="fi-home"></i> Home
  </button>
</p>
<p>Home icon on a green button:
  <button type="button" class="button success">
    <i class="fi-home"></i> Home
  </button>
</p>
<p>Home icon on a large, light blue link button:
  <a href="#" class="button info large">
    <i class="fi-home"></i> Home
  </a>
</p>
```

工具条图标

我们可以使用 `.icon-bar` 类来创建一个指定数量的工具栏图标:

实例

```
<div class="icon-bar five-up">
  <a href="#" class="item">
    <i class="fi-home"></i>
  </a>
  <a href="#" class="item">
    <i class="fi-bookmark"></i>
  </a>
  <a href="#" class="item">
    <i class="fi-info"></i>
  </a>
  <a href="#" class="item">
    <i class="fi-mail"></i>
  </a>
  <a href="#" class="item">
    <i class="fi-like"></i>
  </a>
</div>
```

图标描述使用 `<label>` 元素:

实例

```
<div class="icon-bar five-up">
  <a href="#" class="item">
    <i class="fi-home"></i>
    <label>Home</label>
  </a>
  <a href="#" class="item">
    <i class="fi-share"></i>
    <label>Share</label>
  </a>
  <a href="#" class="item">
    <i class="fi-info"></i>
    <label>Info</label>
  </a>
  <a href="#" class="item">
    <i class="fi-mail"></i>
    <label>Mail</label>
  </a>
  <a href="#" class="item">
    <i class="fi-magnifying-glass"></i>
    <label>Search</label>
  </a>
</div>
```

`.disabled` 类可以让图标变成不可点击状态：

实例

```
<a href="#" class="item disabled">
  <i class="fi-share"></i>
</a>

<a href="#" class="item disabled">
    <i class="fi-mail"></i>
</a>
```

`.vertical` 类用于创建一个垂直的工具栏：

实例

```
<div class="icon-bar vertical five-up">
  ....
</div>
```

Foundation 图标参考手册

更多关于图标的内容，可以参考我们的 Foundation 图标手册。

Foundation 标签

`.label` 类用于提供一些附加信息:

实例

```
<h2>Example <span class="label">New</span></h2>
<h3>Example <span class="label">New</span></h3>
<h4>Example <span class="label">New</span></h4>
```

以下类可以设置标签的颜色: `.secondary` , `.success` , `.info` , `.warning` 或 `.alert` :

实例

```
<span class="label">Default Label</span>
<span class="label secondary">Secondary Label</span>
<span class="label success">Success Label</span>
<span class="label info">Info Label</span>
<span class="label warning">Success Label</span>
<span class="label alert">Alert Label</span>
```

圆角标签

`.radius` 与 `.round` 类可以为标签添加圆角：

实例

```
<span class="label">Default Label</span>
<span class="label radius">Radius Label</span>
<span class="label round">Round Label</span>
<span class="label success round">5</span>
```

标签大小

可以使用 CSS 来修改标签的大小:

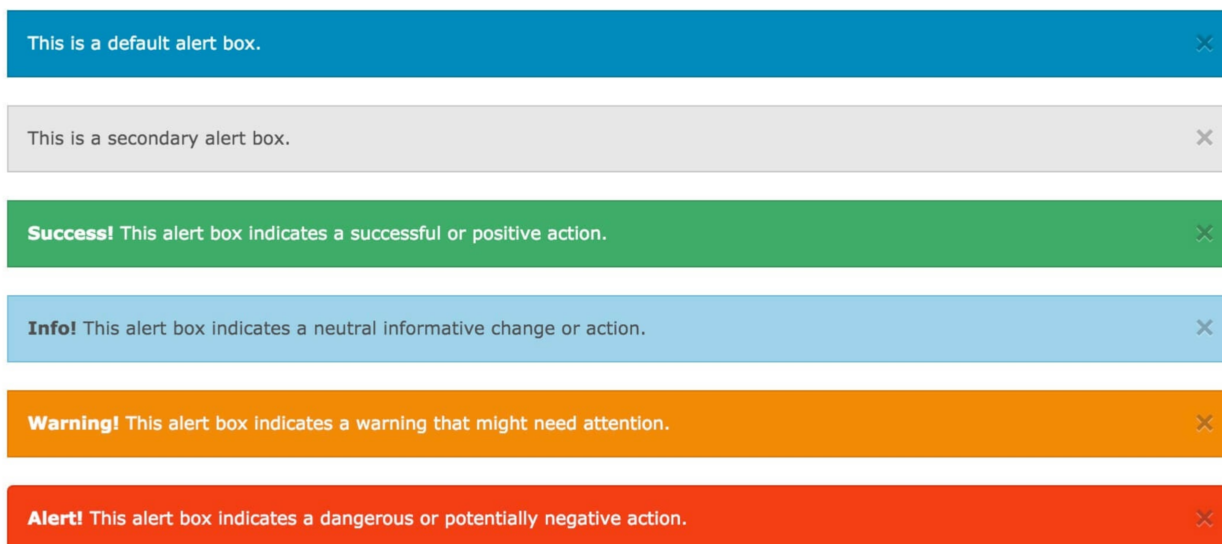
实例

```
<h1>Example <span class="label" style="font-size:36px;">New</span><
<h2>Example <span class="label" style="font-size:30px;">New</span><
<h3>Example <span class="label" style="font-size:20px;">New</span><
<h4>Example <span class="label">New</span></h4>
```



Foundation 提醒框

Foundation 可以很简单的创建一个提醒框：



提醒框可以使用 `.alert-box` 类创建, 可以添加可选的类： `.secondary` , `.success` , `.info` , `.warning` 或 `.alert` :

实例

```
<div data-alert class="alert-box">
  This is a default alert box.
</div>

<div data-alert class="alert-box secondary">
  This is a secondary alert box.
</div>

<div data-alert class="alert-box success">
  <strong>Success!</strong> This alert box indicates a successful c
</div>

<div data-alert class="alert-box info">
  <strong>Info!</strong> This alert box indicates a neutral informa
</div>

<div data-alert class="alert-box warning">
  <strong>Warning!</strong> This alert box indicates a warning that
</div>

<div data-alert class="alert-box alert">
  <strong>Alert!</strong> This alert box indicates a dangerous or p
</div>
```



提醒框的宽度为容器的 100%。

圆角提醒框

`.radius` 和 `.round` 类用于为提醒框添加圆角：

实例

```
<div data-alert class="alert-box success radius">
  <strong>Success!</strong> Alert box with a radius.
</div>

<div data-alert class="alert-box info round">
  <strong>Info!</strong> Alert box that is rounded.
</div>
```

关闭提醒框

要关闭提醒框，可以在连接或按钮元素上添加 `class="close"` 类，并初始化 Foundation JS:

实例

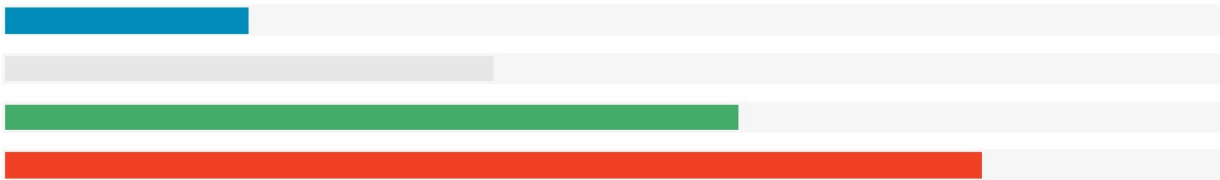
```
<div data-alert class="alert-box">
  This is a default alert box with closing functionality.
  <a href="#" class="close">&times;</a>
</div>

<script>
// Initialize Foundation JS For Functionality
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

> 
>
> × (×) 是一个 HTML 字符实体表示一个关闭按钮的图标，而不是字母 "x"。

Foundation 进度条

Foundation 进度条可以作为程序处理的程度来显示：



我们可以在 `<div>` 元素中使用 `.progress` 类来创建进度条, `.meter` 类用于子元素(<span>)。我们可以在 <span> 元素中设置进度百分比,如下所示：

实例

```
<div class="progress">
  <span class="meter" style="width:70%"></span>
</div>
```

进度条颜色

默认情况下，进度条颜色为蓝色。不同颜色的类为：`.secondary`, `.success`, 或 `.alert`：

实例

```
<div class="progress">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress secondary">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress success">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress alert">
  <span class="meter" style="width:50%"></span>
</div>
```

圆角进度条

`.radius` 和 `.round` 类用于为进度条添加圆角效果：

实例

```
<div class="progress">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress radius">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress rounded">
  <span class="meter" style="width:50%"></span>
</div>
```

进度条尺寸

可以使用 `.small-_num_` 或 `.large-_num_` 来修改进度条的宽度， *num* 是一个数字在 1(8.33%) 和 12(default (100%)) 之间:

实例

```
<div class="progress large-1">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress large-6">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress large-9">
  <span class="meter" style="width:50%"></span>
</div>

<div class="progress large-11">
  <span class="meter" style="width:50%"></span>
</div>

<!-- Default width -->
<div class="progress large-12">
  <span class="meter" style="width:50%"></span>
</div>
```

进度条标签

可以使用 CSS 为进度条添加标签。以下实例中我们添加了<span> 元素来显示百分比：

实例

```
 <style>
.percentage {
  position: absolute;
  top: 50%;
  left: 50%;
  color: white;
  transform: translate(-50%, -50%);
  font-size: 12px;
}
</style>

<div class="progress">
  <span class="meter" style="position:relative;width:75%">
    <span class="percentage">75%</span>
  </span>
</div>

<div class="progress success">
  <span class="meter" style="position:relative;width:50%">
    <span class="percentage">50%</span>
  </span>
</div>

<div class="progress alert">
  <span class="meter" style="position:relative;width:25%">
    <span class="percentage">25%</span>
  </span>
</div>
```

Foundation 面板

Foundation 面板是一个灰色边框，内容含有内边距的模块。可以使用 `.panel` 类来创建:

实例

```
<div class="panel">
  <h3>标题</h3>
  <p>文本内容..</p>
</div>
```

面板颜色

使用 `.callout` 类将面板颜色修改为浅蓝：

实例

```
<div class="panel callout">
  <h3>标题</h3>
  <p>文本内容..</p>
</div>
```

圆角面板

使用 `.radius` 类将面板设置为圆角：

实例

```
<div class="panel radius">
  <h3>标题</h3>
  <p>文本内容..</p>
</div>
```

自定义面板

可以使用 CSS 来自定义面板，以下实例中我们将面板作为一个卡片：

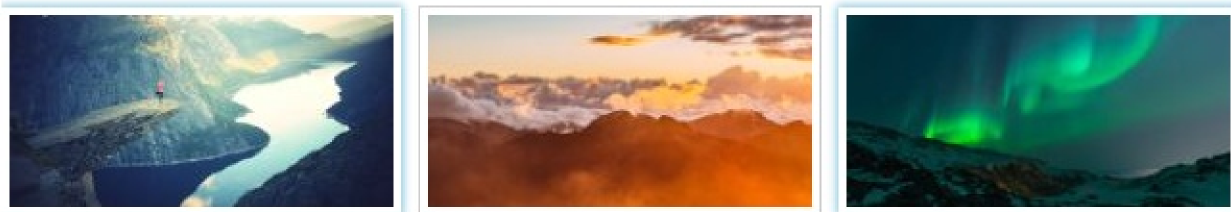
实例

```
<style>
.panel {
    padding: 0;
    border: none;
    width: 50%;
}
div.container {
    text-align: center;
    padding: 15px;
    margin-top: 20px;
}
img {
    width: 100%;
}
</style>

<div class="panel">
  <img src="img/20121204024112919.jpg" alt="Cinque Terre">
  <div class="container">
    <h4>长城</h4>
    <p>不到长城非好汉！！！</p>
  </div>
</div>
```

Foundation 图片

Foundation 提供了响应式的图片，可以创建缩略图喝图片弹窗：



缩略图

在 <img> 元素外添加 <a> 元素将图片作为一个锚链接。

在 <a> 标签中添加 .th 类将图片设置为缩略图。 鼠标移动到上面会显示一个浅蓝色外框:

实例

```
<a href="paris.jpg" class="th">
  <img src="paris.jpg" alt="Paris">
</a>
```



响应式图片

Foundation 中图片默认是响应式的。我们可以在实例页面重置浏览器大小来查看图片缩放效果。

圆角图片

我们可以在 .th 类添加 .radius 类来设置圆角缩略图：

实例

```
<a href="paris.jpg" class="th radius">
  <img src="paris.jpg" alt="Paris">
</a>
```

简洁的弹窗

Foundation 可以很容易实现图片弹窗。

要创建一个弹窗可以在 `<ul>` 元素上添加 `.clearing-thumbs` 类及 `data-clearing` 属性。在 `<ul>` 内添加图片列表。

注意: 图片弹窗需要 JavaScript。所以使用它前需要初始化 Foundation JS。

实例

```
<ul class="clearing-thumbs" data-clearing>
  <li><a href="rock600x400.jpg" class="th"><img src="rock200x100.jp
  <li><a href="skies600x400.jpg" class="th"><img src="skies200x100
  <li><a href="lights600x400.jpg" class="th"><img src="lights200x10
</ul>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

图片文本描述

可以添加 `data-caption` 属性到每个图片来设置图片的描述:

实例

```
<ul class="clearing-thumbs" data-clearing>
  <li><a href="rock600x400.jpg" class="th"><img data-caption="The F
  <li><a href="skies600x400.jpg" class="th"><img data-caption="Sunn
  <li><a href="lights600x400.jpg" class="th"><img data-caption="Nor
</ul>
```



提示： 你可以在 data-caption 属性中添加 HTML 元素，如 data-caption="<h2>Pulpit Rock</h2><p>Located in Norway</p>"

只显示一张缩略图

当你需要实现只显示一张缩略图时你可以在 `<ul>` 中使用 `.clearing-feature` 类并在 `<li>` 中使用 `.clearing-featured-img` 类。

实例

```
<ul class="clearing-thumbs clearing-feature" data-clearing>
  <li><a href="rock600x400.jpg" class="th"><img data-caption="The R
  <li><a href="skies600x400.jpg" class="th"><img data-caption="Sunn
  <li class="clearing-featured-img"><a href="lights600x400.jpg" cla
</ul>
```

Foundation 下拉菜单

Foundation 下拉菜单允许用户从预定义的下拉列表中选取一个值：

实例

```
<!-- Trigger the Dropdown -->
<a href="#" data-dropdown="id01" class="button dropdown">Dropdown B

<!-- Dropdown content -->
<ul id="id01" data-dropdown-content class="f-dropdown">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

实例解析

`.dropdown` 类为按钮添加一个向下的箭头符号"图标。

使用按钮或链接的 `data-dropdown="_id_"` 属性来打开下拉菜单。

id 值需要与下拉菜单的内容 (id01) 匹配。

在 <div>, <nav>, `<ul>` 中添加 `.f-dropdown` 类和 `data-dropdown-content` 属性来创建下拉菜单的内容。

最后初始化 Foundation JS。

注意: 在小屏幕上，所有的下拉菜单的宽度是100%。

下拉菜单尺寸

使用 `.tiny` , `.small` , `.medium` , `.large` 或 `.mega` 来修改下拉菜单的宽度。

注意: 在小屏幕上，所有的下拉菜单的宽度是100%。

实例

```
<!-- Tiny Dropdown: max-width is 200px -->
<ul id="id01" data-dropdown-content class="f-dropdown tiny">..

<!-- Small Dropdown: max-width is 300px -->
<ul id="id02" data-dropdown-content class="f-dropdown small">..

<!-- Medium Dropdown: max-width is 500px -->
<ul id="id03" data-dropdown-content class="f-dropdown medium">

<!-- Large Dropdown: max-width is 800px -->
<ul id="id04" data-dropdown-content class="f-dropdown large">..

<!-- Mega Dropdown: 100% width -->
<ul id="id04" data-dropdown-content class="f-dropdown mega">..
```

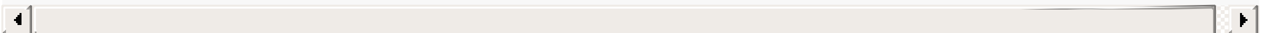
下拉菜单边距

可以使用 `.content` 类为下拉菜单添加内边距：

实例

```
<!-- Default Dropdown -->
<ul id="id01" data-dropdown-content class="f-dropdown">..

<!-- Dropdown with padding -->
<ul id="id02" data-dropdown-content class="f-dropdown content">..
```



其他实例

<div> 下拉菜单中添加多媒体元素：

实例

```
<a href="#" data-dropdown="id01" class="button dropdown">Dropdown B
<div id="id01" data-dropdown-content class="f-dropdown medium conte
  <h4>Paris Title</h4>
  <p>Some text.. some text..</p>
  <img src="paris.jpg" alt="Paris" width="400" height="300">
  <p>Paris, je t'aime.</p>
</div>
```

下拉菜单方向

默认情况下下拉菜单在底部，可以通过添加 `data-options="align:left|right|top"` 来修改其方向：

实例

```
<a href="#" data-dropdown="id01" data-options="align:right" class="
<a href="#" data-dropdown="id02" data-options="align:top" class="bu
<a href="#" data-dropdown="id03" data-options="align:bottom" class=
<a href="#" data-dropdown="id04" data-options="align:left" class="b
```

下拉菜单触发条件

默认情况下，下拉菜单在点击按钮后显示。如果你需要在鼠标移动上去后显示，可以在按钮上使用 `data-options="is_hover:true"` 属性:

实例

```
<a href="#" data-dropdown="id01" data-options="is_hover:true" class
<ul id="id01" data-dropdown-content class="f-dropdown">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>
```

分割按钮

我们可以在按钮上添加 `.split` 类来设置一个分割效果的按钮，分割后会在 <span> 元素上生成一个方向向下的图标按钮:

实例

```
<button class="button split">Split Button
  <span data-dropdown="id01"></span>
</button>

<ul id="id01" data-dropdown-content class="f-dropdown">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
</ul>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

Foundation 折叠列表

在你想隐藏部分内容的显示时，可以使用折叠列表。

实例

```
<ul class="accordion" data-accordion>
  <li class="accordion-navigation">
    <a href="#demo">Simple Collapsible</a>
    <div id="demo" class="content">
      菜鸟教程 -- 学的不仅是技术，更是梦想！！！
    </div>
  </li>
</ul>

<!-- 初始化 Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

实例解析

`.accordion` 类和 `data-accordion` 属性用于创建一个可折叠的元素。`.accordion-navigation` 类用于渲染可折叠元素。实际的内容在 `.content` 类 (<div class="content">) 中，点击按钮既可以显示。

我们在列表项中添加 `<a>` 元素来控制（显示/隐藏）可折叠的内容。然后锚链接使用与可折叠内容内容相同的id (<a href=#demo" 与 <div id="demo"> 相关联)。

注意： 可折叠的效果需要初始化 Foundation JS。

默认情况下，可折叠内容是隐藏的。我们可以通过在 `<div>` 上添加 `.active` 类让其默认是显示的：

实例

```
<div id="demo" class="content active">
```

手风琴效果

手风琴效果用于延展与设置可折叠内容。

手风琴效果通过使用多个不同的锚链接与id来创建：

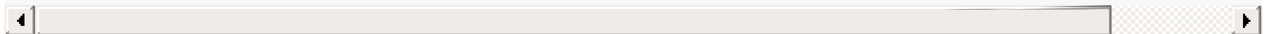
实例

```
<ul class="accordion" data-accordion>
  <li class="accordion-navigation">
    <a href="#demo">手风琴实例 1</a>
    <div id="demo" class="content active">
      Demo 1 - 菜鸟教程 -- 学的不仅是技术，更是梦想！！！
    </div>
  </li>
  <li class="accordion-navigation">
    <a href="#demo2">手风琴实例 2</a>
    <div id="demo2" class="content">
      Demo 2 - Lorem ipsum dolor sit amet....
    </div>
  </li>
  <li class="accordion-navigation">
    <a href="#demo3">手风琴实例 3</a>
    <div id="demo3" class="content">
      Demo 3 - 菜鸟教程 -- 学的不仅是技术，更是梦想！！！
    </div>
  </li>
</ul>
```

默认情况下，手风琴列表项有一个是打开的。如果你想关闭所有可以使用 `data-options="multi_expand:true;"` 属性:

实例

```
<ul class="accordion" data-accordion data-options="multi_expand:tru
  ..
</ul>
```



Foundation 列表

在 HTML 中，无序列表 (`<ul>`) 实例如下:

```
<ul>
  <li>List item</li>
  <li>List item</li>
  <li>List item</li>
  <li>List item</li>
</ul>
```

结果:

- List item
- List item
- List item
- List item

圆圈标识符

在 Foundation 中，无序列表 (`<ul>`) 的前缀符号为圆圈，使用 .circle 类，实例如下:

```
<ul class="circle">
  <li>List item</li>
  ...
</ul>
```

方块标识符

方块标识符前缀使用 .square 类:

```
<ul class="square">
  <li>List item</li>
  ...
</ul>
```

无标识符

如果你不需要标识符，你可以使用 .no-bullet 类 ：

```
<ul class="no-bullet">
  <li>List item</li>
  ...
</ul>
```

如果你需要添加一个水平的列表，可以在 `<ul>` 上添加 `.list-inline` 类：

```
<ul class="inline-list">
```

列表菜单

一些 Web 页面可能需要列表菜单。

在 HTML 中，列表菜单同无序列表 `<ul>` 来定义，例如:

```
<ul>
  <li><a href="#">Home</a></li>
  <li><a href="#">Menu 1</a></li>
  <li><a href="#">Menu 2</a></li>
  <li><a href="#">Menu 3</a></li>
</ul>
```

结果:

- Home
- Menu 1
- Menu 2
- Menu 3

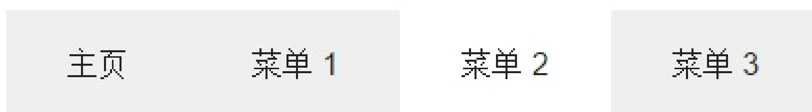
如果你需要添加一个水平的菜单，可以在 `<ul>` 上添加 `.list-inline` 类：

```
<ul class="inline-list">
```

Foundation 选项卡

选项卡导航可以很好的展示不同的内容，并可以对内容进行切换。

切换选项卡



菜单 2

一些文本内容 2

选项卡使用 `<ul class="tabs" data-tab>` 来创建, 各个选项使用 <li> 元素并加上 `.tab-title` 类来创建。

提示: 当前选中项可以使用 `.active` 类。

实例

```
<ul class="tabs" data-tab>
  <li class="tab-title active"><a href="#">Home</a></li>
  <li class="tab-title"><a href="#">Menu 1</a></li>
  <li class="tab-title"><a href="#">Menu 2</a></li>
  <li class="tab-title"><a href="#">Menu 3</a></li>
</ul>
```

垂直的选项卡

垂直选项卡可以使用 `.vertical` 类:

实例

```
<ul class="tabs vertical" data-tab>
```

切换选项卡

如果要设置切换标签，可以使用 <div> 元素加上 `.content` 类。每个选项卡上加上唯一的 ID, 并连接到列表项 (<a href="#menu1" 将打开 id="menu1" 的选项内容)。最后将所有的选项内容放在 `<div>` 元素上，该 `<div>` 元素需要添加 `.tabs-content` 类，并初始化 Foundation JS。

注意 `.active` 类会自动添加到当前选中的选项卡上：

实例

```
<ul class="tabs" data-tab>
  <li class="tab-title active"><a href="#home">Home</a></li>
  <li class="tab-title"><a href="#menu1">Menu 1</a></li>
  <li class="tab-title"><a href="#menu2">Menu 2</a></li>
  <li class="tab-title"><a href="#menu3">Menu 3</a></li>
</ul>
<div class="tabs-content">
  <div class="content active" id="home">
    <h3>HOME</h3>
    <p>Home is where the heart is..</p>
  </div>
  <div class="content" id="menu1">
    <h3>Menu 1</h3>
    <p>Some text, blabla</p>
  </div>
  <div class="content" id="menu2">
    <h3>Menu 2</h3>
    <p>Some other text.</p>
  </div>
  <div class="content" id="menu3">
    <h3>Menu 3</h3>
    <p>Last tab.</p>
  </div>
</div>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

选项卡淡入

使用 CSS 来自定义选项卡淡入的效果：

实例

```
.tabs-content > .content.active {
    -webkit-animation: fadeEffect 1s;
    animation: fadeEffect 1s;
}

@-webkit-keyframes fadeEffect {
    from {opacity: 0;}
    to {opacity: 1;}
}

@keyframes fadeEffect {
    from {opacity: 0;}
    to {opacity: 1;}
}
```

Foundation 分页

如果你的网页有很多内容，就需要使用分页功能。

分页 - 当前页

当前页面需要添加 .current 类:

1 2 3 4 5

要创建一个基础的分页功能需要在 `<ul>` 元素上加上 `.pagination` 类:

实例

```
<ul class="pagination">
  <li><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
</ul>
```

当前页面

可以在 `<li>` 加上 `.current` 类来标注当前页面：

实例

```
<ul class="pagination">
  <li class="current"><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
</ul>
```

禁用分页

如果需要设置某个分页不可点击需要使用 `.unavailable` 类：

实例

```
<ul class="pagination">
  <li><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li class="unavailable"><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
</ul>
```

分页方向

在第一个和最后一个 code> 元素上添加 `.arrow` 类插入 HTML 实体符号 `«` 和 `»` 来创建分页方向符号：

实例

```
<ul class="pagination">
  <li class="arrow"><a href="#">&laquo;</a></li>
  <li><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
  <li class="arrow"><a href="#">&raquo;</a></li>
</ul>
```

分页居中显示

我们可以在 <ul> 外层添加 `<div>` 元素，并在 `<div>` 上添加 `.pagination-centered` 类来实现分页居中显示：

实例

```
<div class="pagination-centered">
  <ul class="pagination">
    <li class="arrow"><a href="#">&laquo;</a></li>
    <li class="current"><a href="#">1</a></li>
    <li><a href="#">2</a></li>
    <li><a href="#">3</a></li>
    <li><a href="#">4</a></li>
    <li><a href="#">5</a></li>
    <li class="arrow"><a href="#">&raquo;</a></li>
  </ul>
</div>
```

面包屑导航

面包屑导航用于展示当前页面的导航结构。

在 `<ul>` 元素上添加 `.breadcrumbs` 类来实现面包屑导航。你可以在 <li> 上添加 `.current` 或 `.unavailable` 类设置当前页与不可点击效果:

实例

```
<ul class="breadcrumbs">
  <li><a href="#">Home</a></li>
  <li><a href="#">Private</a></li>
  <li class="unavailable"><a href="#">Pictures</a></li>
  <li class="current">Vacation</li>
</ul>
```

子导航

在页面切换上，子导航是非常有用的。

在 `<dl>` 元素上添加 `.sub-nav` 类来创建子导航。在 `<dt>` 元素上添加标题，为选中的选项 `<dd>` 添加 `.active` 类:

实例

```
<ul class="sub-nav">
  <dt>Filter:</dt>
  <dd class="active"><a href="#">All</a></dd>
  <dd><a href="#">Active</a></dd>
  <dd><a href="#">Pending</a></dd>
  <dd><a href="#">Suspended</a></dd>
</ul>
```

Foundation 价格表

价格表可用于展示企业产品:

实例

```
<ul class="pricing-table">
  <li class="title">Basic</li>
  <li class="price">$19.99</li>
  <li class="description">Great for small business</li>
  <li class="bullet-item">24/7 Support</li>
  <li class="bullet-item">15GB Storage</li>
  <li class="bullet-item">1GB Bandwidth</li>
  <li class="cta-button"><a class="button" href="#">Buy</a></li>
</ul>
```

实例解析

- `ul.pricing-table` - 定义价格表
- `li.title` - 定义产品标题 (黑色背景)
- `li.price` - 定义价格 (灰色背景字体大个项)
- `li.description` - 定义产品描述 (如果需要)
- `li.bullet-item` - 定义产品特点
- `li.ca-button` - 按钮文本 (如 "Buy", "Join", "Sign Up", 等)

注意: 表格会 100% 填充容器的宽度，所有的项都有边框且是居中的。

价格表网格

以下实例显示了三个企业产品的价格（三项是均等划分宽度的）：

实例

```
<div class="row">
  <div class="medium-4 columns">
    <ul class="pricing-table">
      <li class="title">Basic</li>
      ...
    </ul>
  </div>
  <div class="medium-4 columns">
    <ul class="pricing-table">
      <li class="title">Pro</li>
      ...
    </ul>
  </div>
  <div class="medium-4 columns">
    <ul class="pricing-table">
      <li class="title">Premium</li>
      ...
    </ul>
  </div>
</div>
```

Foundation 顶部导航栏

顶部导航栏放在页面头部：



实例

```
<nav class="top-bar" data-topbar>
  <ul class="title-area">
    <li class="name">
      <!-- 如果你不需要标题或图标可以删掉它 -->
      <h1><a href="#">WebSiteName</a></h1>
    </li>
      <!-- 小屏幕上折叠按钮: 去掉 **.menu-icon** 类, 可以去除图标。
      如果需要只显示图片, 可以删除 "Menu" 文本 -->
    <li class="toggle-topbar menu-icon"><a href="#"><span>Menu</spa
  </ul>

  <section class="top-bar-section">
    <ul class="left">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Page 1</a></li>
      <li><a href="#">Page 2</a></li>
      <li><a href="#">Page 3</a></li>
    </ul>
  </section>
</nav>

<!-- 初始化 Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

实例解析

使用 `<nav class="top-bar" data-topbar>` 创建标准工具条。 `.title-area` 类定义了网站logo区域 (必须防止 `li.name` 内)。屏幕变小后你就可以看到一个 "menu" 按钮。 Foundation 的菜单会根据屏幕尺寸自动折叠喝延展：

小屏幕上，由于尺寸的原因很多选项会被隐藏。 `li.toggle-topbar menu.icon` 类创建了一个菜单的按钮，点击它可以显示被隐藏的选项。 提示: 重置浏览器窗口查看效果。

`.top-bar-section` 定义了导航的链接部分。 `.left` 类指定链接左对齐。 `.active` 类用于显示选中的项，背景为蓝色。

提示: 如果你想导航链接右对齐可以将 `.left` 修改为 `.right` :

实例

```
<section class="top-bar-section">
  <ul class="right">...
```

你可以同时设置左边对齐与右边对齐：

实例

```
<section class="top-bar-section">
  <ul class="left">
    <li class="active"><a href="#">Home</a></li>
    <li><a href="#">Page 1</a></li>
    <li><a href="#">Page 2</a></li>
  </ul>
  <ul class="right">
    <li><a href="#">Sign Up</a></li>
    <li><a href="#">Login</a></li>
  </ul>
</section>
```

导航栏可以通过 `.divider` 类来添加分割线 (大屏幕上是垂直的线，小屏幕上是水平线):

实例

```
<ul class="left">
  <li class="active"><a href="#">Home</a></li>
  <li class="divider"></li>
  <li><a href="#">Page 1</a></li>
  <li class="divider"></li>
  <li><a href="#">Page 2</a></li>
  <li class="divider"></li>
  <li><a href="#">Page 3</a></li>
  <li class="divider"></li>
</ul>
```

导航栏的下拉菜单

顶部导航栏可以设置下拉菜单。

可以通过在 `<li>` 元素上添加 `.has-dropdown` 类来设置下拉菜单:

实例

```
<section class="top-bar-section">
  <ul class="left">
    <li class="active"><a href="#">Home</a></li>
    <li class="has-dropdown">
      <a href="#">Dropdown</a>
      <ul class="dropdown">
        <li><a href="#">First link in dropdown</a></li>
        <li><a href="#">Second link in dropdown</a></li>
        <li class="active"><a href="#">Active link in dropdown</a><
      </ul>
    </li>
  </ul>
</section>
```

分割线

使用 `.divider` 类来设置下拉菜单的分割线:

实例

```
<ul class="dropdown">
  <li><a href="#">Apple</a></li>
  <li><a href="#">Banana</a></li>
  <li><a href="#">Orange</a></li>
  <li class="divider"></li>
  <li><a href="#">Kale</a></li>
  <li><a href="#">Spinach</a></li>
</ul>
```

下拉菜单标签

在 `<li>` 内添加 `<label>` 元素来设置下拉菜单的标签(标题):

实例

```
<ul class="dropdown">
  <li><label>Fruit</label></li>
  <li><a href="#">Apple</a></li>
  <li><a href="#">Banana</a></li>
  <li><a href="#">Orange</a></li>
  <li class="divider"></li>
  <li><label>Vegetable</label></li>
  <li><a href="#">Kale</a></li>
  <li><a href="#">Spinach</a></li>
</ul>
```

内嵌下拉菜单

下拉菜单可以再嵌入一个下拉菜单：

实例

```
<section class="top-bar-section">
  <ul class="left">
    <li class="has-dropdown">
      <a href="#">Dropdown</a>
      <ul class="dropdown">
        <li><label>Level 1</label></li>
        <li><a href="#">Link</a></li>
        <li><a href="#">Link</a></li>
        <li class="has-dropdown">
          <a href="#">New dropdown</a>
          <ul class="dropdown">
            <li><label>Level 2</label></li>
            <li><a href="#">2nd level dropdown</a></li>
            <li><a href="#">2nd level dropdown</a></li>
            <li class="has-dropdown">
              <a href="#">New dropdown</a>
              <ul class="dropdown">
                <li><label>Level 3</label></li>
                <li><a href="#">3rd level dropdown</a></li>
                <li><a href="#">3rd level dropdown</a></li>
              </ul>
            </li>
          </ul>
        </li>
      </ul>
    </li>
  </ul>
</section>
```

可点击

默认情况下导航栏的下拉菜单在鼠标移动过去后显示，我们可以使用 `data-options="is_hover: false"` 属性来设置导航栏在鼠标在点击后显示:

实例

```
<nav class="top-bar" data-topbar data-options="is_hover: false">
```

导航栏上的按钮及图标

你可以在导航栏上放置图标和按钮：

实例

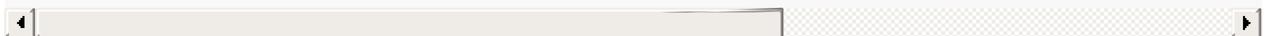
```
<li><a href="#" class="button">Button Link</a></li>
```

你可以在导航栏上放上图标，更多图片样式可以查看 Foundation 图标教程:

实例

```
<head>
<!-- Foundation 图标样式 -->
<link rel="stylesheet" href="http://static.runoob.com/assets/founda
</head>

<ul class="left">
  <li class="active"><a href="#"><i class="fi-home"></i> Home</a></
  <li><a href="#"><i class="fi-torso"></i> Sign Up</a></li>
  <li><a href="#"><i class="fi-magnifying-glass"></i> Search</a></l
</ul>
```



固定导航栏

导航栏可以固定在页面顶部。

页面滚动时导航栏在顶部是不会动的。

要固定导航栏只需要将导航栏放在 `<div class="fixed">` 内即可:

实例

```
<div class="fixed">
  <nav class="top-bar" data-topbar>
    ...
  </nav>
</div>
```

导航栏绝对定位

我们可以将导航栏放在 `<div class="sticky">` 内来设置导航栏的绝对定位，当滚动条滚到到该区域时，该导航栏就像固定导航栏一样在顶部不动:

实例

```
<div class="sticky">
  <nav class="top-bar" data-topbar>
    ...
  </nav>
</div>
```

当你使用 `.sticky` 类时，顶部导航栏在所有屏幕尺寸上将固定不动。如果你需要在指定屏幕上设定只需要在 `<nav>` 上添加 `data-options="sticky_on: small|medium|large"` 属性即可：

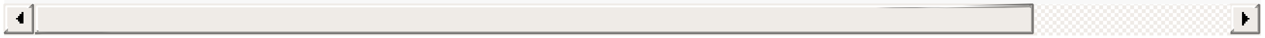
实例

```
<div class="sticky">
  <!-- 只有在大屏幕上 -->
  <nav class="top-bar" data-topbar data-options="sticky_on: large">
    ..
   </nav>
</div>
```

或者通过数组设置多个屏幕尺寸：

实例

```
<div class="sticky">
  <!-- 小屏幕和大屏幕 (没有中等屏幕)-->
  <nav class="top-bar" data-topbar data-options="sticky_on: [small,
    ..
   </nav>
</div>
```



Foundation 侧边栏

侧边栏导航

Foundation 使用 `<ul class="side-nav">` 创建侧边栏:

实例

```
<ul class="side-nav">
  <li><a href="#">Link 1</a></li>
  <li><a href="#">Link 2</a></li>
  <li><a href="#">Link 3</a></li>
  <li><a href="#">Link 4</a></li>
</ul>
```

激活链接与分割线

已点击的链接可以在 `<li>` 上使用 `.active` 类来标识，使用 `.divider` 类添加水平分割线：

实例

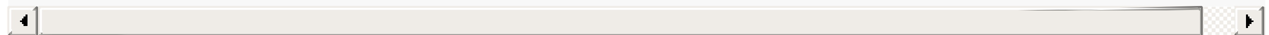
```
<ul class="side-nav">
  <li class="active"><a class="a" href="#">Link 1</a></li>
  <li><a class="a" href="#">Link 2</a></li>
  <li class="divider"></li>
  <li><a class="a" href="#">Link 3</a></li>
  <li><a class="a" href="#">Link 4</a></li>
</ul>
```

网格中的侧边栏

我们可以使用网格设计模式来设置侧边导航栏，实例如下：

实例

```
<div class="row">
  <div class="medium-4 columns" style="background-color:#f1f1f1;">
    <ul class="side-nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">Learn HTML</a></li>
      ...
    </ul>
  </div>
  <div class="medium-8 columns">
    <h1>Side Nav</h1>
    <p>Some text..</p>
    ...
  </div>
</div>
```



Foundation 滑动导航(Off-Canvas)

侧边栏导航

Off-Canvas 滑动导航现在逐渐在移动页面变得越来越流行了 (点击菜单按钮菜单从左侧滑出):



创建滑动导航

创建滑动导航实例如下：

实例

```
<!-- 最外层div：页面布局 -->
<div class="off-canvas-wrap" data-offcanvas>
  <!-- 内部元素: "工具栏" 内容 (图标, 链接, 描述内容等)-->
  <div class="inner-wrap">
    <nav class="tab-bar">
      <section class="left-small">
        <a class="left-off-canvas-toggle menu-icon" href="#"><span>
      </section>

      <section class="middle tab-bar-section">
        <h1 class="title">Off-canvas Example</h1>
      </section>
    </nav>

    <!-- 滑动菜单 -->
    <aside class="left-off-canvas-menu">
      <!-- Add links or other stuff here -->
      <ul class="off-canvas-list test">
        <li><label>Heading</label></li>
        <li><a href="#">Link 1</a></li>
        <li><a href="#">Link 2</a></li>
        ...
      </ul>
    </aside>

    <!-- 主要内容 -->
    <section class="main-section">
      <h3>Lorem Ipsum</h3>
      <p>....</p>
    </section>

    <!-- 关闭菜单 -->
    <a class="exit-off-canvas"></a>

  </div> <!-- 结束内部内容 -->
</div> <!-- 结束滑动菜单 -->

<!-- 初始化 Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

Foundation 麦哲伦（Magellan）导航

Magellan

麦哲伦导航就是一个导航索引，位置是固定的，会根据用户滚动当前展示的内容自动切换导航栏上的导航项:

Page 1 Page 2

如何创建麦哲伦导航

麦哲伦导航就是一个导航索引，创建方式如下:

实例

```
<div data-magellan-expedition="fixed">
  <dl class="sub-nav">
    <dd data-magellan-arrival="page1"><a href="#page1">Page 1</a></
    <dd data-magellan-arrival="page2"><a href="#page2">Page 2</a></
  </dl>
</div>

<h3 data-magellan-destination="page1">Page1</h3>
<a name="page1"></a>
...

<h3 data-magellan-destination="page2">Page2</h3>
<a name="page2"></a>
...

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

实例解析

在 <div> 元素上添加 `data-magellan-expedition="fixed"` 属性来创建麦哲伦导航。

然后在 `<dd>` 或 `<li>` 上添加 `data-magellan-arrival="_value_"` 属性，后面添加一个与该属性值一样的链接(page1)。

使用 `data-magellan-destination="value"` 属性来控制麦哲伦导航的目标, 后面紧跟的 `<a>` 元素添加 `name="_value_"` 属性。两个属性的值必须与 `data-magellan-arrival` 的值一致 (**page1**)。

最后，初始化 Foundation JS，用户在滚动页面时导航就会根据当前显示的内容自动切换。

麦哲伦导航头部工具条

麦哲伦导航使用头部工具条实例：

实例

```
<div data-magellan-expedition="fixed">
  <nav class="top-bar" data-topbar>
    ...

    <section class="top-bar-section">
      <ul class="left">
        <li data-magellan-arrival="page1"><a href="#page1">Page 1</
        <li data-magellan-arrival="page2"><a href="#page2">Page 2</
      </ul>
    </section>

  </nav>
</div>

<h3 data-magellan-destination="page1">Page1</h3>
<a name="page1"></a>
...

<h3 data-magellan-destination="page2">Page2</h3>
<a name="page2"></a>
...
```

麦哲伦导航内边距

默认情况下，麦哲伦导航的 `<div>` 元素有 10px 的内边距。可以使用 CSS 移除它：

实例

```
[data-magellan-expedition], [data-magellan-expedition-clone] {
    padding: 0;
}
```

麦哲伦导航选项

使用 data-options 属性修改麦哲伦导航的设置, 例如 `<div data-magellan-expedition="fixed" data-options="destination_`
:

名称	类型	默认	描述	实例
active_class	string	active	指定激活链接的类	
threshold	number	0	指定导航在什么时候需要固定位置。会根据滚动条滚动计算，默认为 0 (auto)。	
destination_threshold	number	20	设该值设定了导航链接显示为激活（蓝色背景）时导航列表距离顶部的值。	
fixed_top	number	0	指定了导航条距离头部的像素值	

Foundation 表单

Foundation 表单控制会自动设置为全局样式:

所有 `<textarea>` , `<select>` 及 `<input>` 元素宽度都为 100%，且带有外边距、内边距、阴影喝鼠标移动效果。

实例

```
<form>
  Input:
  <input type="text" placeholder="Name">

  Textarea:
  <textarea rows="4" placeholder="Address"></textarea>

  Select:
  <select>
    <option>1</option>
    <option>2</option>
    <option>3</option>
    <option>4</option>
  </select>
</form>
```

标签

在表单中使用 `<label>` 元素来设置标签，标签可以添加 for 属性和 id 属性。用户在点击标签或输入域时获取输入框焦点：

实例

```
<form>
  <label for="name">Input
    <input type="text" placeholder="Name" id="name">
  </label>

  <label for="adr">Label
    <textarea rows="4" placeholder="Address" id="adr"></textarea>
  </label>

  <label for="num">Select
    <select id="num">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
    </select>
  </label>
</form>
```

如果需要设置标签右对齐，可以使用 `.right` 类:

实例

```
<label class="right">
```

Fieldset

Foundation 渲染 `<fieldset>` 元素的样式如下：

实例

```
<form>
  <fieldset>
    <legend>Fieldset Legend</legend>
    <label>Name
      <input type="text" placeholder="First Name..">
    </label>
    <label>Email
      <input type="text" placeholder="Enter email..">
    </label>
  </fieldset>
</form>
```

错误状态

使用 `.error` 类来设置错误的标签、输入框、文本框样式:

实例

```
<form>
  <label class="error">Error
    <input type="text" placeholder="Name..">
  </label>
  <small class="error">Wrong input</small>

  <textarea rows="4" placeholder="Address"></textarea>
  <small class="error">Wrong input</small>
</form>
```



> 你需要使用 JavaScript 来更新用户输入的错误状态。

Foundation 输入框尺寸

First Name

Name

Last Name

Last Name

City

City

Search

Search | Go

使用网格的列来设置输入框的大小，如 `.large-6` , `.medium-6` , 等。

更多网格系统知识，可以点击 网格系统 教程。

实例

```
<form>
  <div class="row">
    <div class="large-10 medium-7 columns">
      <label>large-10 medium-7 (100% on small)
        <input type="text" placeholder="Name">
      </label>
    </div>
  </div>

  <div class="row">
    <div class="small-5 columns">
      <label>small-5
        <input type="text" placeholder="Name">
      </label>
    </div>
  </div>

  <div class="row">
    <div class="medium-3 columns">
      <label>medium-3 (100% on small)
        <input type="text" placeholder="Name">
      </label>
    </div>
  </div>
</form>
```

相等大小列

以下演示了相等大小列的实例:

实例

```
<form>
  <div class="row">
    <div class="medium-4 columns">
      <label>medium-4 (100% on small, stacked)
        <input type="text" placeholder="Name">
      </label>
    </div>

    <div class="medium-4 columns">
      <label>medium-4 (100% on small, stacked)
        <input type="text" placeholder="Name">
      </label>
    </div>

    <div class="medium-4 columns">
      <label>medium-4 (100% on small, stacked)
        <input type="text" placeholder="Name">
      </label>
    </div>
  </div>
</form>
```

内联标签

如果你希望你的标签内容显示在左边（不是上边），可以将标签元素 label 放在输入框左边的不同的列上，并使用 `.inline` 类来设置垂直居中:

实例

```
<form>
  <div class="row">
    <div class="small-8">
      <div class="row">
        <div class="small-3 columns">
          <label for="name" class="inline right">Name</label>
        </div>
        <div class="small-9 columns">
          <input type="text" id="name" placeholder="First Name..">
        </div>
      </div>
    </div>
  </div>
</form>
```

前置和后置标签

你可以在 `<div class="row collapse">` 中添加前置和后置标签，元素为： `<_element_ class="postfix">` 或 `<_element_ class="prefix">` 。可以使用网格系统来设置前置和后置标签的大小：

实例

```
<form>
  <div class="row">
    <div class="large-6 columns">
      <div class="row collapse prefix-radius">
        <div class="small-3 columns">
          <span class="prefix">Prefix</span>
        </div>
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
      </div>
    </div>
    <div class="large-6 columns">
      <div class="row collapse postfix-radius">
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
        <div class="small-3 columns">
          <span class="postfix">Postfix</span>
        </div>
      </div>
    </div>
  </div>
</form>
```

前置和后置标签按钮

可以使用 `<a>` 元素添加 `.button` 类来设置前置和后置按钮:

实例

```
<a href="#" class="postfix button">Go</a>
```

前置和后置标签圆角按钮

实例

```
<form>
  <div class="row">
    <div class="large-6 columns">
      <div class="row collapse prefix-radius">
        <div class="small-3 columns">
          <span class="prefix">Prefix</span>
        </div>
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
      </div>
    </div>
    <div class="large-6 columns">
      <div class="row collapse postfix-radius">
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
        <div class="small-3 columns">
          <span class="postfix">Postfix</span>
        </div>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="large-6 columns">
      <div class="row collapse prefix-round">
        <div class="small-3 columns">
          <a href="#" class="button prefix">Go</a>
        </div>
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
      </div>
    </div>
    <div class="large-6 columns">
      <div class="row collapse postfix-round">
        <div class="small-9 columns">
          <input type="text" placeholder="Value">
        </div>
        <div class="small-3 columns">
          <a href="#" class="button postfix">Go</a>
        </div>
      </div>
    </div>
  </div>
</form>
```

Foundation 开关

开关是在鼠标点击(手指敲击)下在 "On" 和 "Off" 状态下切换：



切换开关使用 `<div class="switch">` 创建。 `<div>` 内添加带有唯一 id 的 `<input type="checkbox">` ， `<label>` 元素的 for 属性需要与 `<input type="checkbox">` 的 id 相匹配：

实例

```
<div class="switch">
  <input id="mySwitch" type="checkbox">
  <label for="mySwitch"></label>
</div>
```

开关大小

使用 `.large` , `.small` 或 `.tiny` 类来设置开关大小：

实例

```
<div class="switch large">...</div>
<div class="switch">...</div>
<div class="switch small">...</div>
<div class="switch tiny">...</div>
```

圆角切换开关

使用 `.radius` 和 `.round` 类来设置圆角切换开关：

实例

```
<div class="switch">...</div>
<div class="switch radius">...</div>
<div class="switch round">...</div>
```

开关组

可以通过设置单选按钮(radio)来设置单个选项。注意: 注意各个选项的 `name` 属性必须一致 (实例中为 "myGroup")。

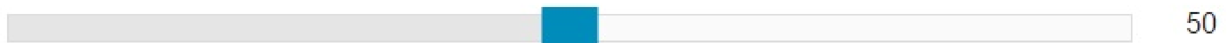
实例

```
<div class="switch">
  <input id="mySwitch1" type="radio" name="myGroup">
  <label for="mySwitch1"></label>
</div>

<div class="switch">
  <input id="mySwitch2" type="radio" name="myGroup" checked>
  <label for="mySwitch2"></label>
</div>
```

Foundation 滑块

Foundation 滑块允许用户通过拖动来选取区间值:



滑块可以通过使用 <div class="range-slider" data-slider> 创建。在 <div> 内, 添加两个 <span> 元素: <span class="range-slider-handle"> 创建矩形滑块（蓝色背景）， <span class="range-slider-active-segment"> 是在滑块后的灰色横条，是滑块拖动区域。

注意: 滑块需要使用 JavaScript。所以你需要初始化 oundation JS:

实例

```
<div class="range-slider" data-slider>
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

圆角和禁用滑块

使用 `.radius` 和 `.round` 类来添加圆角滑块。使用 `.disabled` 类来禁用滑块：

实例

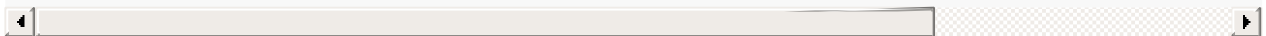
```
<div class="range-slider" data-slider>..</div>
<div class="range-slider radius" data-slider>...</div>
<div class="range-slider round" data-slider>...</div>
<div class="range-slider disabled" data-slider>...</div>
```

垂直滑块

使用 `.vertical-range` 类和 `data-options="vertical: true;"` 来创建垂直滑块:

实例

```
<div class="range-slider vertical-range" data-slider data-options='
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```

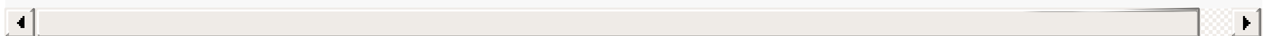


滑块值

默认情况下，滑块放在横条的中间 (数值为 "50")。可以通过添加 `data-options="initial: _num_"` 属性来修改默认值:

实例

```
<div class="range-slider" data-slider data-options="initial: 80;">
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```



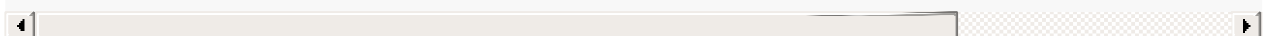
显示滑块值

如果我们需要在滑块拖动时实时显示值，可以通过在 `<div>` 中添加 `data-options="display_selector:#_id_"` 属性且元素 id 值与滑块的 id 匹配，如下实例：

实例

```
<!-- Display the slider value in this span -->
<span id="mySlider"></span>

<div class="range-slider" data-slider data-options="display_selecto
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```

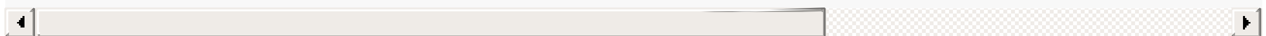


组合数据选项

以下实例使用了 `display_selector` 和 `initial` 数据选项:

实例

```
<span id="mySlider"></span>
<div class="range-slider" data-slider data-options="display_selecto
 <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```

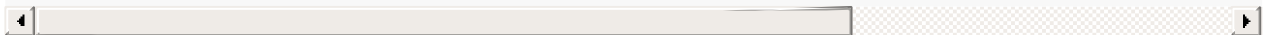


步长

默认情况下，滑块滑动的增加减少的数量为 "1"。可以通过添加 `data-options="step: _num_;"` 属性来修改步长值。实例中设置为 25:

实例

```
<span id="mySlider"></span>
<div class="range-slider" data-slider data-options="display_selecto
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```

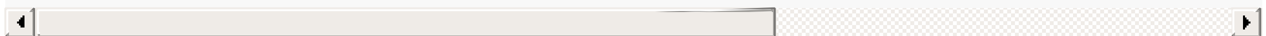


自定义区间

默认情况下，区间值在 "0" 到 "100"。可以通过添加 data-options `start` 和 `end` 来设置区间值。以下实例设置区间值为 "1" 到 "20":

实例

```
<span id="mySlider"></span>
<div class="range-slider" data-slider data-options="display_selecto
  <span class="range-slider-handle"></span>
  <span class="range-slider-active-segment"></span>
</div>
```

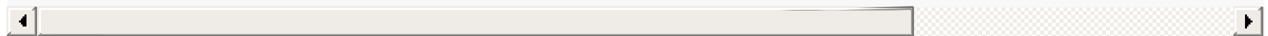


使用网格

以下使用为在网格中使用滑块：

实例

```
<div class="row">
  <div class="small-10 columns">
    <div class="range-slider" data-slider data-options="display_sel
      <span class="range-slider-handle"></span>
      <span class="range-slider-active-segment"></span>
    </div>
  </div>
  <div class="small-2 columns">
    <!-- The display element (Tip: use CSS to perfectly position it
    <span id="mySlider" style="display:block;margin-top:14px;"></sp
  </div>
</div>
```

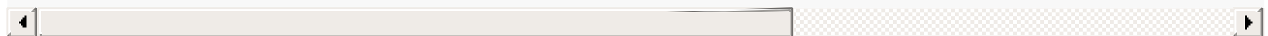


使用 Input

以下实例使用 `<input>` 取代 `<span>` 来显示滑块的值:

实例

```
<div class="row">
  <div class="small-10 columns">
    <div class="range-slider" data-slider data-options="display_sel
      <span class="range-slider-handle"></span>
      <span class="range-slider-active-segment"></span>
    </div>
  </div>
  <div class="small-2 columns">
    <!-- The display element (Tip: use CSS to perfectly position it
    <input type="number" id="mySlider" style="margin-top:7px;" valu
  </div>
</div>
```



Foundation 提示框

提示框在鼠标移动到元素上后显示：

鼠标移到我这!　　我这也能显示!

我们可以在任何元素上添加 `data-tooltip` 属性来创建提示信息。使用 `title` 属性来设置提示信息的文本。

注意: 滑块需要使用 JavaScript。所以你需要初始化 oundation JS:

实例

```
<span data-tooltip title="Hooray!">Hover over me!</span>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

`.has-tip` 类可以加粗鼠标移动的文本:

实例

```
<span data-tooltip class="has-tip" title="Hooray!">Hover over me!<
```

提示框显示位置

默认情况下，提示框会出现在元素的底部。

可以使用 `.tip-top` , `.tip-left` , `.tip-right` or `.tip-bottom` (默认) 来设置提示框的位置。

注意: 在小尺寸的屏幕上，提示框的宽带是 100%。

实例

```
<span data-tooltip class="has-tip tip-top" title="Hooray!">Top</spa
<span data-tooltip class="has-tip tip-bottom" title="Hooray!">Botto
<span data-tooltip class="has-tip tip-left" title="Hooray!">Left</s
<span data-tooltip class="has-tip tip-right" title="Hooray!">Right<
```

圆角提示框

`.radius` 和 `.round` 类用于设置圆角提示框：

实例

```
<span data-tooltip class="has-tip" title="Hooray!">Default</span>
<span data-tooltip class="has-tip radius" title="Hooray!">Radius</s
<span data-tooltip class="has-tip round" title="Hooray!">Round</spa
```

Foundation 模态框

模态框是一个显示在页面头部的弹窗。

我们可以在容器元素上(如 `<div id="myModal"`)使用唯一 ID，并添加 `.reveal-modal` 类和 `data-reveal` 属性来添加模态框。我们可以在任何元素上使用 `data-reveal-id="_id_"` 属性阿里打开模态框。*id* 必须与容器 id 一致 (实例为 "myModal")。

如果你希望在点击模态框之外的区域来关闭模态框。你可以在模态框的关闭按钮 `<a>` 标签上添加 `.close-reveal-modal` 类来实现。

注意: 滑块需要使用 JavaScript。所以你需要初始化 oundation JS:

实例

```
<!-- Trigger the Modal -->
<button type="button" class="button" data-reveal-id="myModal">Click

<!-- The Modal Content -->
<div id="myModal" class="reveal-modal" data-reveal>
  <h2>Heading in Modal.</h2>
  <p>Some text in the modal.</p>
  <p>Some text in the modal.</p>
  <a class="close-reveal-modal">&times;</a>
</div>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

模态框大小

可以在模态框容器上添加以下类来设置模态框的大小：

- `.tiny` : 30% 宽度
- `.small` : 40% 宽度
- `.medium` : 60% 宽度
- `.large` : 70% 宽度
- `.xlarge` : 95% 宽度
- `.full` : 100% 宽度和高度

注意: 在平板、笔记本、PC 电脑上默认为 80% 宽度，在小屏幕设备上是 100% 宽度。

实例

```
<div id="myModal" class="reveal-modal tiny|small|medium|large|xlarg
```

内嵌模态框

你可以在模态框内嵌入模态框，可以在第一个模态框上添加新的触发按钮。你必须为内嵌模态框设置一个唯一的 id：

实例

```
<!-- Trigger the modal -->
<a href="#" class="button" data-reveal-id="myModal">Click To Open M

<!-- The First Modal -->
<div id="myModal" class="reveal-modal" data-reveal>
  <h2>First Modal</h2>
  <p>Some text..</p>
  <p><a href="#" data-reveal-id="secondModal" class="button">Open S
  <a class="close-reveal-modal">&times;</a>
</div>

<!-- The Second Modal -->
<div id="secondModal" class="reveal-modal" data-reveal>
  <h2>Tada! Second Modal</h2>
  <p>Some text..</p>
  <a class="close-reveal-modal">&times;</a>
</div>
```

第二个模态框会取代第一个模态框。如果你希望在打开第二个模态框时，不关闭第一个模态框。可以在第二个模态框上添加 `data-options="multiple_opened:true;"` 属性：

实例

```
<div id="secondModal" class="reveal-modal" data-reveal data-options
```

Foundation Joyride

Joyride 是一个功能向导的 JavaScript 效果，创建实例如下：

实例

```
<!-- Elements that control the tour stops -->
<h3 id="first">First stop!</h3>
<h3 id="second">Second stop!</h3>

<!-- The joyride: must be placed at the bottom of your page, but i
<ol class="joyride-list" data-joyride>
  <li data-id="first">
    <p>First stop. The ride has begun!</p>
  </li>
  <li data-id="second">
    <h4>Second Stop</h4>
    <p>Any valid HTML will work inside the Joyride.</p>
  </li>
  <li data-button="End">
    <h4>End Stop</h4>
    <p>The tour is over. You can either go back to the previous sto
  </li>
</ol>

<!-- Start Joyride Upon Initialization -->
<script>
$(document).ready(function() {
    $(document).foundation('joyride', 'start');
})
</script>
```

实例解析

以上实例中，我们创建了两个元素，每个元素都有独立的 ID。 两个元素设置了 joyride 开始和结束的位置。

我们在 `<ol>` 或 `<ul>` 元素上添加 `data-joyride` 属性和 `.joyride-list` 类来创建 joyride。你需要在文档头部定义它 (在 `<body>` 内的头部)。在每个列表上使用 `<li>` 元素，每个元素添加 `data-id="_value_"` 属性。属性的 *value* 必须与之前元素的 id 相同。所以第一个功能导航 `<h3>` 元素使用 id="first" 必须与 <li> 元素的 data-id="first" 值一致。

如果你没有管理停止的 id，将显示一个模态框。

最后，Joyride 需要使用 JavaScript 初始化它，代码为：

```
$(document).foundation('joyride', 'start');
```

Foundation 均衡器(Equalizer)

我们可以在容器元素添加 `data-equalizer` 属性，并为每个子元素添加 `data-equalizer-watch` 属性来创建一个相同高度的均衡器。最高的元素决定了其他元素的高度。

注意: 滑块需要使用 JavaScript。所以你需要初始化 oundation JS:

实例

```
<div class="row" data-equalizer>
  <div class="medium-4 columns panel" data-equalizer-watch>
    Lorem ipsum...
  </div>
  <div class="medium-4 columns panel" data-equalizer-watch>
    Sed ut...
  </div>
  <div class="medium-4 columns panel" data-equalizer-watch>
    Lorem ipsum...
  </div>
</div>

<!-- Initialize Foundation JS -->
<script>
$(document).ready(function() {
    $(document).foundation();
})
</script>
```

不同屏幕的均衡器

在均衡器上通过添加 `data-equalizer-mq="_value_"` 属性为不同屏幕尺寸设置相同高度。值可以是以下之一：

值	描述	实例
`medium-up`	默认。 创建相同高度的容器， 宽度大于 40.063em	
`large-up`	创建相同高度的容器， 宽度大于 64.063em	
`xlarge-up`	创建相同高度的容器， 宽度大于 90.063em	
`xxlarge-up`	创建相同高度的容器， 宽度大于 120.063em	

嵌套内容

为 `data-equalizer` 和 `data-equalizer-watch` 属性添加相同的值。 这会一起连接到均衡器容器。 重复多次嵌套均衡器，确保他们是匹配的：

实例

```
<!-- The Equalized Container -->
<div class="row" data-equalizer="first">
  <div class="medium-4 columns">
    <div class="panel" data-equalizer-watch="first">
      <h3>Panel</h3>

      <!-- An Equalized Container Inside The Equalized Container --
      <div class="row" data-equalizer="second">
        <div class="panel" data-equalizer-watch="second">
          <h3>Nested Panel</h3>
          <p>Lorem ipsum...</p>
        </div>
        <div class="panel" data-equalizer-watch="second">
          <h3>Nested Panel</h3>
          <p>Lorem ipsum...</p>
        </div>
        <div class="panel" data-equalizer-watch="second">
          <h3>Nested Panel</h3>
          <p>Lorem ipsum...</p>
        </div>
      </div>
      <!-- End Nested Equalized Container -->

    </div>
  </div>
  <div class="medium-4 columns">
    <div class="panel" data-equalizer-watch="first">
      <h3>Panel</h3>
      <p>Ut enim...</p>
    </div>
  </div>
  <div class="medium-4 columns">
    <div class="panel" data-equalizer-watch="first">
      <h3>Panel</h3>
      <p>Lorem ipsum....</p>
    </div>
  </div>
</div>
```

Foundation 网格

Foundation 网格系统

Foundation 网格系统为 12 列。

如果你不需要 12 列，你可以合并一些列，创建一些更大宽度的列。

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1
span 4				span 4				span 4			
span 4				span 8							
span 6						span 6					
span 12											

Foundation 的网格系统是响应式的。 列会根据屏幕尺寸自动调整大小。在大尺寸屏幕上，可能是三列，小屏幕尺寸就可能是三个单列，按顺序排列。

网格列

Foundation 网格系统有三个列：

- `.small` (手机端)
- `.medium` (平板设备)
- `.large` (电脑设备：笔记本，台式机)

以上类可以结合使用，创建更灵活的布局

基本的网格结构

以下是基本的 Foundation 网格结构实例:

实例

```
<div class="row">
  <div class="small|medium|large-_num_ columns"></div>
</div>
<div class="row">
  <div class="small|medium|large-_num_ columns"></div>
  <div class="small|medium|large-_num_ columns"></div>
  <div class="small|medium|large-_num_ columns"></div>
</div>
<div class="row">
  ...
</div>
```

首先，创建一行 (`<div class="row">`)。 这是一个水平的垂直列。然后添加列的数量说明 `small-_num_` ， `medium-_num_` 及 `large-_num_` 类。注意：列的数量 `_num_` 加起来必须等于 12 :

实例

```
<div class="row">
  <div class="small-12 columns">.small-12 yellow</div>
</div>
<div class="row">
  <div class="small-8 columns">.small-8 beige</div>
  <div class="small-4 columns">.small-4 gray</div>
</div>
<div class="row">
  <div class="large-9 small-8 columns">.small-8 .large-9 pink</div
  <div class="large-3 small-4 columns">.small-4 .large-3 orange</d
</div>
```

实例中，第一行的 <div> 类为 `.small-12` , 这会创建 12 列（100%宽度）。

第二行创建了两个列， `.small-4` 的宽度为33.3% ， `.small-8` 的宽度为66.6%。

第三行我们添加了额外的两个列 (`.large-3` 和 `.large-9`)。这意味着如果在大屏幕尺寸下，列就会变为 25% (`.large-3`) 和 75% (`.large-9`)的比例。同时我们也指定了小屏幕上列的比例 33% (`.small-4`) 和 66% (`.small-8`) 。这种组合的方式对于不同屏幕显示效果是非常有帮助的。

网格选项

下表总结了 Foundation 网格系统在多个设备上的说明：

	小型设备	中等设备	大设备
	Phones (<40.0625em (640px))	Tablets (>=40.0625em (640px))	Laptops & Desktops (>=64.0625em (1025px))
网格行为	一直是水平的	以折叠开始，断点以上是水平的	以折叠开始，断点以上是水平的
类前缀	.small-*	.medium-*	.large-*
类的数量	12	12	12
可内嵌	Yes	Yes	Yes
偏移量	Yes	Yes	Yes
列排序	Yes	Yes	Yes

宽屏

网格最大(`.row`) 宽度为 62.5rem。在宽屏上，当宽度大于 62.5rem, 列不会跨越页面的宽度， 即使宽度设定为 100%。但你可以通过 CSS 重新设置 max-width:

实例

```
<style>
.row {
    max-width: 100%;
}
</style>
```

如果你使用默认的 max-width, 但希望背景颜色跨越整个页面宽度，你可以使用 `.row` 包裹整个容器，并指定你需要的背景颜色:

实例

```
<div style="background-color:coral;padding:25px;">
  <div class="row">
    <div class="small-6 columns" style="background-color:yellow;">.
    <div class="small-6 columns" style="background-color:pink;">.sm
  </div>
</div>
```



Foundation 网格 - 水平堆叠

以下实例创建了一个基本的网格系统，在 PC 和平板设备上它是水平平铺的，在手机等小型设备上它是水平堆叠的。

实例

```
<div class="row">
  <div class="medium-6 columns" style="background-color:yellow;">
    <p>菜鸟教程</p>
  </div>
  <div class="medium-6 columns" style="background-color:pink;">
    <p>菜鸟教程</p>
  </div>
</div>
```



提示: `.small|medium|large-*` 类中的数字指定了跨越的列数。所以, `.small-1` 跨越 1 列, `.small-4` 跨越 4 列, `.small-6` 跨越 6 列 (50% 宽度) 等。

注意: 要保证数列加起来是 12 列!

Foundation 网格 - 小型设备

假设我们在小型设备上有一个简单的网格布局，两列，宽度比例为 25% 和 75%。

提示: 小型设备的定义是屏幕小于 `40.0625em` 。

小型设备上我们使用 `.small-*` 类。

```
<div class="small-3 columns">....</div>
<div class="small-9 columns">....</div>
```

以下实例设置了两个列，比例为 25% 和 75% (Foundation 是移动优先: 如果没有特别说明，在大型设备上会继承 .small 类的代码): `.small` in them and use those".

实例

```
<div class="row">
  <div class="small-3 columns" style="background-color:yellow;">
    <p>菜鸟教程</p>
  </div>
  <div class="small-9 columns" style="background-color:pink;">
    <p>菜鸟教程</p>
  </div>
</div>
```



注意: 要保证数列加起来是 12 列!

如果需要设置 33.3%/66.6% 的比例，你可以使用 `.small-4` 和 `.small-8` :

实例

```
<div class="row">
  <div class="small-4 columns" style="background-color:yellow;">
    <p>菜鸟教程</p>
  </div>
  <div class="small-8 columns" style="background-color:pink;">
    <p>菜鸟教程</p>
  </div>
</div>
```

Foundation 网格 - 中型设备

上一章节我们介绍了小型设备上我们使用 `.small-*` 类来设置，网格比例为 25%/75%：

```
<div class="small-3 columns">....</div>
<div class="small-9 columns">....</div>
```

在中型设备上我们推荐的比例为 50%/50%。

提示: 中型设备的屏幕尺寸定义在 `40.0625em` 到 `64.0624em` 之间。

中型设备上使用 `.medium-*` 类。

现在我们在中型设备上添加两列：

```
<div class="small-3 medium-6 columns">....</div>
<div class="small-9 medium-6 columns">....</div>
```

以上实例设置了两个列，比例为 25% 和 75% (Foundation 是移动优先: 如果没有特别说明，在大型设备上会继承 .small 类的代码):

小型设备上使用的比例为 25%/75% (`.small-3` 和 `.small-9`)。但在中型设备上使用的比例为 50%/50% (`.medium-6` 和 `.medium-6`) 。

实例

```
<div class="row">
  <div class="small-3 medium-6 columns" style="background-color:yel
    <p>菜鸟教程</p>
  </div>
  <div class="small-9 medium-6 columns" style="background-color:pir
    <p>菜鸟教程</p>
  </div>
</div>
```



注意: 要保证数列加起来是 12 列!

紧在中型设备上使用

以下实例中我们指定了 `.medium-6` 类 (不是 `.small-*`)。这表明在中型或大型设备上比例为 50%/50%。但在小型设备上会水平堆叠 (100% 宽度):

实例

```
<div class="row">
  <div class="medium-6 columns" style="background-color:yellow;">
    <p>菜鸟教程</p>
  </div>
  <div class="medium-6 columns" style="background-color:pink;">
    <p>菜鸟教程</p>
  </div>
</div>
```

Foundation 网格 - 大型设备

上一章节我们介绍了中型设备和小型设备的网格布局，小型设备上使用的比例为 25%/75% (.small-3 和 .small-9)，但在中型设备上使用的比例为 50%/50% (.medium-6 和 .medium-6)：

```
<div class="small-3 medium-6 columns">....</div>
<div class="small-9 medium-6 columns">....</div>
```

在大型设备上我们推荐的比例为 33%/66%。

提示: 大型设备的屏幕尺寸定义大于 64.0625em。

大型设备上使用 `.large-*` 类。

现在我们在大型设备上添加两列：

```
<div class="small-3 medium-6 large-4 columns">....</div>
<div class="small-9 medium-6 large-8 columns">....</div>
```

解析

- 小型设备两个列的比例为 25%/75% (`.small-3` 和 `.small-9`)
- 中型设备两个列的比例为 50%/50% (`.medium-6` 和 `.medium-6`)
- 大型设备两个列的比例为 33%/66% (`.large-4` 和 `.large-8`)

实例

```
<div class="row">
  <div class="small-3 medium-6 large-4 columns" style="background-c
    <p>菜鸟教程</p>
  </div>
  <div class="small-9 medium-6 large-8 columns" style="background-c
    <p>菜鸟教程</p>
  </div>
</div>
```



注意: 要保证数列加起来是 12 列!

紧在大型设备上使用

以下实例中我们指定了 `.large-6` 类 (不是 .medium- 和 `*.small-*`)。这表明在大型设备上比例为 50%/50%。但在中型或小型设备上会水平堆叠 (100% 宽度):

实例

```
<div class="row">
  <div class="large-6 columns" style="background-color:yellow;">
    <p>菜鸟教程</p>
  </div>
  <div class="large-6 columns" style="background-color:pink;">
    <p>菜鸟教程</p>
  </div>
</div>
```

Foundation 块状网格

块状网格用于均分页面内容：例如一行内要显示四张图片，不管什么屏幕下都需要均分宽度。

可以使用 `<ul>` 元素加上 `.small|medium|large-block-grid-_num_` 类来创建块状网格。*num* 用于指定均分是数量：

实例

```
<ul class="small-block-grid-3">
  <li><img src="newyork.jpg" alt="New York"></li>
  <li><img src="paris.jpg" alt="Paris"></li>
  <li><img src="sanfran.jpg" alt="San Francisco"></li>
</ul>
```

另一个实例，使用段落：

实例

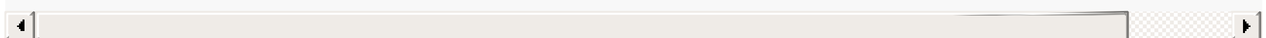
```
<ul class="small-block-grid-3">
  <li><p>Just a Simple Example Text...</p></li>
  <li><p>Just a Simple Example Text...</p></li>
  <li><p>Just a Simple Example Text...</p></li>
</ul>
```

不同尺寸屏幕显示不同数量

通过添加多个网格块类可以设置不同尺寸屏幕显示不同的块数量:

实例

```
<ul class="small-block-grid-2 medium-block-grid-3 large-block-grid-
  <li><img src="newyork.jpg" alt="New York"></li>
  <li><img src="paris.jpg" alt="Paris"></li>
  <li><img src="sanfran.jpg" alt="San Francisco"></li>
  <li><img src="newyork.jpg" alt="New York"></li>
</ul>
```



Foundation 网格实例

以下我们收集了一些网格常用的实例。

三个均等列

该实例演示了如何创建三个均等列 (33.3%/33.3%/33.3%)，在中型和大型设备上显示三个列，在小型设备上自动堆叠：

实例

```
<div class="row">
  <div class="medium-4 columns" style="background-color:yellow;">
    <p>.medium-4</p>
  </div>
  <div class="medium-4 columns" style="background-color:pink;">
    <p>.medium-4</p>
  </div>
  <div class="medium-4 columns" style="background-color:yellow;">
    <p>.medium-4</p>
  </div>
</div>
```

三个不均等列

该实例演示了如何创建三个不均等列 (25%/50%/25%)，在中型和大型设备上显示三个列，在小型设备上自动堆叠：

实例

```
<div class="row">
  <div class="medium-3 columns" style="background-color:yellow;">
    <p>.medium-3</p>
  </div>
  <div class="medium-6 columns" style="background-color:pink;">
    <p>.medium-6</p>
  </div>
  <div class="medium-3 columns" style="background-color:yellow;">
    <p>.medium-3</p>
  </div>
</div>
```

两个均等列

该实例演示了如何创建两个均等列 (50%/50%)，在小型、中型和大型设备上列的比例始终为 50%/50%：

实例

```
<div class="row">
  <div class="small-6 columns" style="background-color:yellow;">
    <p>.small-6</p>
  </div>
  <div class="small-6 columns" style="background-color:pink;">
    <p>.small-6</p>
  </div>
</div>
```

两个不均等列

该实例演示了如何创建两个不均等列 (33.3%/66.6%)，在小型、中型和大型设备上列的比例始终为 33.3%/66.6%：

实例

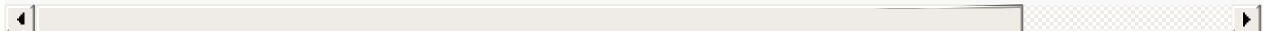
```
<div class="row">
  <div class="small-8 columns" style="background-color:yellow;">
    <p>.small-8</p>
  </div>
  <div class="small-4 columns" style="background-color:pink;">
    <p>.small-4</p>
  </div>
</div>
```

修改列的顺序

通过使用 `.small|medium|large-push-*` 和 `.small|medium|large-pull-*` 类来修改列的顺序:

实例

```
<div class="row">
  <div class="small-4 small-8-push columns" style="background-colo
    <p>.small-4 .small-8-push</p>
  </div>
  <div class="small-8 small-4-pull columns" style="background-colo
    <p>.small-8 .small-4-pull</p>
  </div>
</div>
```



嵌套列

你可以使用嵌套网格(列中插入列):

实例

```
<div class="row">
  <div class="small-8 columns">.small-8
    <div class="row">
      <div class="small-8 columns">.small-8 Nested
        <div class="row">
          <div class="small-8 columns">.small-8 Nested Again</div>
          <div class="small-4 columns">.small-4</div>
        </div>
      </div>
      <div class="small-4 columns">.small-4</div>
    </div>
  </div>
  <div class="small-4 columns">.small-4</div>
</div>
```

混合：手机、桌面设备

Foundation 网格系统有三个列: `.small-*` (手机), `.medium-*` (平板), 和 `.large-*` (桌面设备)。这些类可以动态组合使用，让布局更加灵活：

提示： 每个类都能放大，如果你希望小型和大型屏幕设备的宽度一样可以设置指定 `.small-*` 。

实例

```
<div class="row">
  <div class="small-6 large-8 columns">.small-6 .large-8</div>
  <div class="small-6 large-4 columns">.small-6 .large-4</div>
</div>
<div class="row">
  <div class="small-2 large-4 columns">.small-2 .large-2</div>
  <div class="small-4 large-4 columns">.small-4 .large-2</div>
  <div class="small-6 large-4 columns">.small-6 .large-2</div>
</div>
<div class="row">
  <div class="small-3 large-5 columns">.small-3 .large-5</div>
  <div class="small-9 large-7 columns">.small-9 .large-7</div>
</div>
```

混合：手机、平板和桌面设备

实例

```
<div class="row">
  <div class="medium-6 large-8 columns">.medium-6 .large-8</div>
  <div class="medium-6 large-4 columns">.medium-6 .large-4</div>
</div>
<div class="row">
  <div class="small-4 medium-3 large-7 columns">.small-4 .medium-3
  <div class="small-4 medium-6 large-3 columns">.small-4 .medium-6
  <div class="small-4 medium-3 large-2 columns">.small-4 .medium-3
</div>
```

居中列

列居中可以使用 `.small-centered` 类。中型和大型设备可以继承小型设备的居中，但你需要在大型设备上设置居中类 `.large-centered` 。

实例

```
<div class="row">
  <div class="small-4 small-centered columns">small-4 small-centere
</div>
<div class="row">
  <div class="small-6 small-centered columns">small-6 small-centere
</div>
<div class="row">
  <div class="small-6 large-centered columns">small-6 large-centere
</div>
<div class="row">
  <div class="small-8 small-centered large-uncentered columns">smal
</div>
<div class="row">
  <div class="small-10 small-centered columns">small-10 small-cente
</div>
```

列偏移量

可以使用 `.large-offset-*` (或 `.small-offset-*`) 类设置列向右移。 左侧外边距的列数量使用 * 号控制:

实例

```
<div class="row">
  <div class="large-1 columns">1</div>
  <div class="large-11 columns">11</div>
</div>
<div class="row">
  <div class="large-1 columns">1</div>
  <div class="large-10 large-offset-1 columns">10, offset 1</div>
</div>
<div class="row">
  <div class="large-1 columns">1</div>
  <div class="large-9 large-offset-2 columns">9, offset 2</div>
</div>
<div class="row">
  <div class="large-1 columns">1</div>
  <div class="large-8 large-offset-3 columns">8, offset 3</div>
</div>
```

不完整列

如果一行中的列数量之和不是 12 , Foundation 将自动将最后一列向右浮动，并使用空白来填充剩下的列。

可选项 `.end` 类用于设置最后一列的元素向左边浮动:

实例

```
<div class="row">
  <div class="medium-3 columns">.medium-3</div>
  <div class="medium-3 columns">.medium-3</div>
  <div class="medium-3 columns">.medium-3</div>
</div>
<div class="row">
  <div class="medium-3 columns">.medium-3</div>
  <div class="medium-3 columns">.medium-3</div>
  <div class="medium-3 columns end">.medium-3 .end</div>
</div>
```

宽屏

网格 (`.row`) 最大尺寸（ max-width）为 62.5rem。在宽屏设备上尺寸可能大于 62.5rem, 这样列就无法完整填充页面，即便宽度设置为 100%。但是我们可以通过 CSS 来设置新的 max-width:

实例

```
<style>
.row {
    max-width: 100%;
}
</style>
```

如果你想使用默认的 max-width, 但是背景颜色需要跨域整个页面，这时你在容器元素上使用 `.row` 类，并指定你需要的背景颜色:

实例

```
<div style="background-color:coral;padding:25px;">
  <div class="row">
    <div class="small-6 columns" style="background-color:yellow;">
    <div class="small-6 columns" style="background-color:pink;">.sr
  </div>
</div>
```

Foundation 参考手册

Foundation 图标参考手册

Foundation 图标

Foundation 提供了 283 图标，你可以使用 CSS 来渲染她们：修改大小和颜色。

图标创建语法格式如下：

```
<i  class="fi-name"></i>
```

语法中 *name* 部分为图标指定的名称。

要使用图标，你可以将图片的样式文件放置在页面头部 `<head>` 部分:

```
<link  rel="stylesheet"  href="http://static.runoob.com/assets/foun
```

以上是菜鸟教程使用的图标样式类，大家可以共用。

点我在新页面中查看图标。

General Icons

- heart
- star
- plus
- minus
- x
- check
- upload
- download
- widget
- marker
- refresh
- home
- trash
- paperclip
- lock
- unlock
- calendar
- cloud
- magnifying-glass

- zoom-out
- zoom-in
- wrench
- rss
- share
- flag
- list-thumbnails
- list
- thumbnails
- annotate
- folder
- folder-lock
- folder-add
- clock
- play-video
- crop
- archive
- pencil
- graph-trend
- graph-bar
- graph-horizontal
- graph-pie
- checkbox
- minus-circle
- x-circle
- eye
- database
- results
- results-demographics
- like
- dislike
- upload-cloud
- camera
- alert
- bookmark
- contrast
- mail
- video
- telephone
- comment
- comment-video
- comment-quotes
- comment-minus
- comments
- microphone
- megaphone
- sound
- address-book

- bluetooth
- html5
- css3
- layout
- web
- foundation

Page Icons

- page
- page-csv
- page-doc
- page-pdf
- page-export
- page-export-csv
- page-export-doc
- page-export-pdf
- page-add
- page-remove
- page-delete
- page-edit
- page-search
- page-copy
- page-filled
- page-multiple

Arrow Icons

- arrow-up
- arrow-right
- arrow-down
- arrow-left
- arrows-out
- arrows-in
- arrows-expand
- arrows-compress

People Icons

- torso
- torso-female
- torsos
- torsos-female-male
- torsos-male-female
- torsos-all

- torsos-all-female
- torso-business

Device Icons

- monitor
- laptop
- tablet-portrait
- tablet-landscape
- mobile
- mobile-signal
- usb

Text Editor Icons

- bold
- italic
- underline
- strike
- text-color
- background-color
- superscript
- subscript
- align-left
- align-center
- align-right
- align-justify
- list-numbered
- list-bullet
- indent-more
- indent-less
- print
- save
- photo
- filter
- paint-bucket
- link
- unlink
- quote

Media Control Icons

- play
- stop
- pause

- previous
- rewind
- fast-forward
- next
- record
- play-circle
- volume-none
- volume
- volume-strike
- loop
- shuffle
- eject
- rewind-ten

Ecommerce Icons

- dollar
- euro
- pound
- yen
- bitcoin
- bitcoin-circle
- credit-card
- shopping-cart
- burst
- burst-new
- burst-sale
- paypal
- price-tag
- pricetag-multiple
- shopping-bag
- dollar-bill

Accessibility Icons

- wheelchair
- braille
- closed-caption
- blind
- asl
- hearing-aid
- guide-dog
- universal-access
- telephone-accessible
- elevator
- male

- female
- male-female
- male-symbol
- female-symbol

Social & Brand Icons

- social-500px
- social-adobe
- social-amazon
- social-android
- social-apple
- social-behance
- social-bing
- social-blogger
- social-delicious
- social-designer-news
- social-deviant-art
- social-digg
- social-dribbble
- social-drive
- social-dropbox
- social-evernote
- social-facebook
- social-flickr
- social-forrst
- social-foursquare
- social-game-center
- social-github
- social-google-plus
- social-hacker-news
- social-hi5
- social-instagram
- social-joomla
- social-lastfm
- social-linkedin
- social-medium
- social-myspace
- social-orkut
- social-path
- social-picasa
- social-pinterest
- social-rdio
- social-reddit
- social-skype
- social-skillshare
- social-smashing-mag

- social-snapchat
- social-spotify
- social-squidoo
- social-stack-overflow
- social-steam
- social-stumbleupon
- social-treehouse
- social-tumblr
- social-twitter
- social-vimeo
- social-windows
- social-xbox
- social-yahoo
- social-yelp
- social-youtube
- social-zerply
- social-zurb

Miscellaneous Icons

- compass
- music
- lightbulb
- battery-full
- battery-half
- battery-empty
- projection-screen
- info
- power
- asterisk
- at-sign
- key
- ticket
- book
- book-bookmark
- anchor
- puzzle
- foot
- paw
- mountains
- trees
- sheriff-badge
- first-aid
- trophy
- prohibited
- no-dogs
- no-smoking

- safety-cone
- shield
- crown
- target
- target-two
- die-one
- die-two
- die-three
- die-four
- die-five
- die-six
- skull
- map
- clipboard
- clipboard-pencil
- clipboard-notes

Foundation CSS 参考手册

Foundation 默认设置

Foundation 使用浏览器默认字体大小 (`font-size:100%`)。对于大多数桌面设备的浏览器来说，字体大小默认为 16px。对于移动设备的浏览器，字体默认大小为 12px。 默认的字体为 `"Helvetica Neue"` ， line-height 默认为 `1.5` 。

这些设置是适用于 `<body>` 元素内的元素。

此外， `<p>` 元素与底部的外边距(margin-bottom) 为 1.25rem , line-height 为 1.6。

文本

以下的 HTML 元素，Foundation 设置了独立的样式渲染它，不会采用浏览器默认样式。点击 "尝试一下" 查看在线实例。

元素	描述	在线实例
<h1> - <h6>	h1 - h6 标题	
<a>	浅蓝色的链接，鼠标移动到链接会有下划线	
<small>	浅灰色的副标题文本	
<blockquote>	引用内容模块	
<strong>	加粗文本	
<em>	斜体	
<abbr>	指定单词的缩写，使用该元素文本出现虚线下划线，鼠标移动上去会有提示信息	
<kbd>	接收键盘输入指令: <kbd>CTRL + P</kbd>	
<hr>	水平线	
<code>	代码片段	
	无序列表	
	有序列表	
<dl>	描述性列表	

文本对齐

使用 CSS 类来修改文本的对齐方式：

类	描述	实例
.text-left	左对齐文本	
.text-right	右对齐文本	
.text-center	居中	
.text-justify	两端对齐	

不同尺寸屏幕的对齐

使用 CSS 类来修改文本的不同尺寸屏幕的对齐方式：

类	描述	实例
左对齐		
.small-text-left	所有尺寸屏幕左对齐	
.small-only-text-left	小尺寸屏幕左对齐(宽度小于 40em)	
.medium-text-left	宽度大于 40.0625em 尺寸屏幕左对齐	
.medium-only-text-left	宽度在 40.0625em 到 64em 尺寸的屏幕左对齐	
.large-text-left	宽度大于 64.0625em 尺寸屏幕左对齐	
.large-only-text-left	宽度在 64.0625em 到 90em 尺寸的屏幕左对齐	
.xlarge-text-left	宽度大于 90.0625em 尺寸屏幕左对齐	
.xlarge-only-text-left	宽度在 90.0625em 到 120em 尺寸的屏幕左对齐	
.xxlarge-text-left	宽度大于 120em 尺寸屏幕左对齐	
右对齐		
.small-text-right	所有尺寸屏幕右对齐	
.small-only-text-right	小尺寸屏幕右对齐(宽度小于 40em)	
.medium-text-right	宽度大于 40.0625em 尺寸屏幕右对齐	
.medium-only-text-	宽度在 40.0625em 到 64em 尺寸的屏幕右对	

right	齐
.large-text-right	宽度大于 64.0625em 尺寸屏幕右对齐
.large-only-text-right	宽度在 64.0625em 到 90em 尺寸的屏幕右对齐
.xlarge-text-right	宽度大于 90.0625em 尺寸屏幕右对齐
.xlarge-only-text-right	宽度在 90.0625em 到 120em 尺寸的屏幕右对齐
.xxlarge-text-right	宽度大于 120em 尺寸屏幕右对齐
居中对齐	
.small-text-center	所有尺寸屏幕居中对齐
.small-only-text-center	小尺寸屏幕居中对齐(宽度小于 40em)
.medium-text-center	宽度大于 40.0625em 尺寸屏幕居中对齐
.medium-only-text-center	宽度在 40.0625em 到 64em 尺寸的屏幕居中对齐
.large-text-center	宽度大于 64.0625em 尺寸屏幕居中对齐
.large-only-text-center	宽度在 64.0625em 到 90em 尺寸的屏幕居中对齐
.xlarge-text-center	宽度大于 90.0625em 尺寸屏幕居中对齐
.xlarge-only-text-center	宽度在 90.0625em 到 120em 尺寸的屏幕居中对齐
.xxlarge-text-center	宽度大于 120em 尺寸屏幕居中对齐
两端对齐	
.small-text-justify	所有尺寸屏幕都两端对齐
.small-only-text-justify	小尺寸屏幕两端对齐(宽度小于 40em)
.medium-text-justify	宽度大于 40.0625em 尺寸屏幕两端对齐
.medium-only-text-justify	宽度在 40.0625em 到 64em 尺寸的屏幕两端对齐
.large-text-justify	宽度大于 64.0625em 尺寸屏幕两端对齐
.large-only-text-justify	宽度在 64.0625em 到 90em 尺寸的屏幕两端对齐
.xlarge-text-justify	宽度大于 90.0625em 尺寸屏幕两端对齐
.xlarge-only-text-justify	宽度在 90.0625em 到 120em 尺寸的屏幕两端对齐

.xxlarge-text-justify	宽度大于 120em 尺寸屏幕两端对齐

其他

类	描述	实例
.left	元素向左浮动	
.right	元素向右浮动	
.clearfix	清除浮动 - 必须添加在浮动元素的父元素上	
.hide	隐藏元素 (CSS `display: none`)	
.list-inline	将所有元素设置在同一行	
.lead	让 <p> 元素更突出	
.subheader	设置浅色的 <h1> - <h6> 元素	

Foundation CSS 可见性

根据屏幕尺寸显示元素

以下类会根据设备(屏幕尺寸)来显示元素。

类	描述
.show-for-small-only	只在小型设备上显示元素 (屏幕宽度小于 40.0625em)
.show-for-medium-up	在中型及以上设备上显示元素 (屏幕宽度大于 40.0625em)
.show-for-medium-only	只在中型设备上显示元素 (屏幕宽度在 40.0625em 到 64.0625em 之间)
.show-for-large-up	在大型及以上设备上显示元素 (屏幕宽度大于 64.0625em)
.show-for-large-only	只在大型设备上显示元素 (屏幕宽度在 64.0625em 到 90.0625em 之间)
.show-for-xlarge-up	在更大型及以上设备上显示元素 (屏幕宽度大于 90.0625em)
.show-for-xlarge-only	只在更大型及以上设备上显示元素 (屏幕宽度在 90.0625em 到 120.0625em之间)
.show-for-xxlarge-up	在超大型及以上设备上显示元素 (屏幕宽度大于 120.0625em)

以下实例演示了以上所有 `.show-` 类的可见性。

```
<p class="show-for-small-only">你在小型设备上。</p>
<p class="show-for-medium-up">你在中型、大型、更大型、超大型的设备上。</p
<p class="show-for-medium-only">你在中型设备上。</p>
<p class="show-for-large-up">你在大型、更大型、超大型的设备上</p>
<p class="show-for-large-only">你在大型设备上。</p>
<p class="show-for-xlarge-up">你在更大型、超大型的设备上。</p>
<p class="show-for-xlarge-only">你在更大型设备上。</p>
<p class="show-for-xxlarge-up">你在超大型设备上。</p>
```

根据屏幕尺寸隐藏元素

以下类会根据设备(屏幕尺寸)来隐藏元素。

类	描述
.hide-for-small-only	只在小型设备上隐藏元素 (屏幕宽度小于 40.0625em)
.hide-for-medium-up	在中型及以上设备上隐藏元素 (屏幕宽度大于 40.0625em)
.hide-for-medium-only	只在中型设备上隐藏元素 (屏幕宽度在 40.0625em 到 64.0625em 之间)
.hide-for-large-up	在大型及以上设备上隐藏元素 (屏幕宽度大于 64.0625em)
.hide-for-large-only	只在大型设备上隐藏元素 (屏幕宽度在 64.0625em 到 90.0625em 之间)
.hide-for-xlarge-up	在更大型及以上设备上隐藏元素 (屏幕宽度大于 90.0625em)
.hide-for-xlarge-only	只在更大型及以上设备上隐藏元素 (屏幕宽度在 90.0625em 到 120.0625em之间)
.hide-for-xxlarge-up	在超大型及以上设备上隐藏元素 (屏幕宽度大于 120.0625em)

```
<p class="hide-for-small-only">你不在小型设备上。</p>
<p class="hide-for-medium-up">你不在中型、大型、更大型、超大型的设备上。</
<p class="hide-for-medium-only">你不在中型设备上。</p>
<p class="hide-for-large-up">你不在大型、更大型、超大型的设备上。</p>
<p class="hide-for-large-only">你不在大型设备上。</p>
<p class="hide-for-xlarge-up">你不在更大型、超大型的设备上。</p>
<p class="hide-for-xlarge-only">你不在更大型设备上。</p>
<p class="hide-for-xxlarge-up">你不在超大型设备上。</p>
```

根据屏幕方向显示元素

以下类会根据设备(屏幕尺寸)来隐藏元素。

我们可以设置元素在不同方向是是否显示或隐藏。笔记本等桌面设备一般是横向的，但是手机和平板设备可以是横向或纵向，我们可以根据用户手机拿的方向来设置元素隐藏与显示：

类	描述
.show-for-landscape	在横向时显示元素（纵向隐藏）
.show-for-portrait	在纵向时显示元素（横向隐藏）

下面实例根据使用的方向显示文本内容：

实例

```
<p class="show-for-landscape">文本只在横向显示。</p>
<p class="show-for-portrait">文本只在纵向显示。</p>
```

触屏设备的显示与隐藏

你可以根据设备是否支持触摸来显示与隐藏元素。

类	描述
.show-for-touch	在支持触屏的设备上显示(不支持的设备上隐藏)
.hide-for-touch	在支持触屏的设备上隐藏(不支持的设备上显示)

下面实例根据设备是否支持触摸来显示文本内容：

实例

```
<p class="show-for-touch">你的设备支持触屏。</p>
<p class="hide-for-touch">你的设备不支持触屏。</p>
```